Photonic Three-Level Systems Within the Framework of Generalised Probabilistic Theories

by

Michael Grabowecky

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Science
in
Physics - (Quantum Information)

Waterloo, Ontario, Canada, 2021

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

If one seeks to test quantum theory not against a particular alternative but against many alternatives in a landscape of possibilities, it is crucial to be able to analyze experimental data in a theory-agnostic way. This can be achieved using the framework of Generalized Probabilistic Theories (GPTs). A GPT identifies the properties of a physical theory based entirely on what it predicts for the probabilities associated with the various outcomes of measurements in an experiment. The use of the GPT formalism allows one to avoid any biases towards quantum theory that typically arise in conventional practices. In this work, we provide an experimental characterization of a photonic three-level system within the framework of GPTs. First, we outline the relevant background information in quantum mechanics and experimental quantum optics. We then review the basics of the GPT framework and describe how to apply a particular tomographic scheme to our data that we use to characterize our experimental three-level system. This scheme achieves a GPT characterization of each of the preparations and measurements implemented in the experiment without requiring any prior characterization of either. Finally, we perform the experiment and analyze the data to obtain GPT characterizations of our experimentally realized state and effect vector spaces.

Our experimental three-level system is encoded in a single photon shared among three modes which are distinguished by their polarization and spatial degrees of freedom. Our analysis identifies that the most likely dimension of the GPT vector space for the realized three-level system is 9, agreeing with the value predicted by quantum theory. We infer the scope of GPTs that are consistent with our experimental data and identify pairs of bounding polytopes of the state and effect spaces, whose shapes resemble those of the quantum mechanical state and effect spaces for a qutrit. From these spaces, we are able to place quantitative bounds on possible deviations from quantum theory. In particular, we bound the degree to which the no-restriction hypothesis might be violated for our three-level system.

Acknowledgements

My time at the University of Waterloo has been challenging on both personal and intellectual levels. Additionally, the coronavirus pandemic imposed further challenges both on the advancement of my research as well as my mental health. For these reasons, it is very important for me to acknowledge the many individuals that made my time at UWaterloo so great, despite these hardships.

To begin, I want to thank my supervisor Kevin Resch. Without the guidance and leadership that you have provided me throughout my masters, the experiment presented in this thesis and the results it revealed would not have materialized. I would also like to thank Robert Spekkens for the many meetings and interesting discussions relating to my research. Thank you to Elie Wolfe for sharing your immense knowledge of linear programming with me. Thank you to Michal Bajcy and Thomas Jennewein for agreeing to join my defence committee.

Thank you to all the current and past group members of QOQI: Andrew Cameron, Sandra Cheng, Patrick Daley, Sacha Schwarz, Benjamin Maclellan, and Ruoxuan Xu. I would particularly like to thank Andrew Cameron for all the help you gave when I asked countless questions and all the time you spent helping me think through problems. I couldn't have done this without you.

Thank you to all current and past QOQI co-op students: Christopher Pollack, Tim Hill, Benjamin Wong, Jack Bishop, Aldo Pasos, and Satchel Armena. I especially want to thank Christopher Pollack for the enormous amount of work invested into this project both during his time as a co-op student in QOQI as well as his time at the University of Toronto.

Additionally, I want to thank Yasmeen Kamaliddin. I could not have made it through the past two years without you. I'm so fortunate to have you in my life.

Finally, I would like to thank my incredible family: Cindy, Chris, Cheryl, Oliver, Abby, Bev, and Gerry for their constant love and support.

Dedication

To my grandmother, Geraldine Hartley. May you rest in peace.

Table of Contents

List of Figures

List of Tables

Chapter 1

Geometry of Quantum Theory

A great introduction to quantum information theory can be found in Refs. [32] and [40]. Sections 1.1 and 1.2 loosely follow these references. Section 1.1 outlines some of the basics of the quantum formalism such as states, measurements, and operations. Section 1.2 outlines the geometry of qubit quantum mechanics. Specifically, the quantum state and measurement spaces for a qubit system is derived. Section 1.3 extends this derivation to three-level (qutrit) quantum systems. Finally, section 1.4 discusses the relevance of the Haar measure in quantum mechanics and how it can be used to generate a truly uniform distribution of random n-dimensional quantum states.

1.1 Quantum Mechanics

1.1.1 States

In all areas of physics, a state is typically thought of as an attribute of a physical object. For example, a state can designate an object's position, momentum, energy, or any other measurable quantity. In practice, a state is defined much more broadly, containing all information for how a system interacts with itself, other systems and its surrounding environment. In quantum mechanics, a state is an element of a complex vector space in which an inner product is well defined. Such a vector space is know as a *Hilbert space*.

Pure quantum states are denoted as $|\psi\rangle$ and take the form of a d-dimensional column vector, where d is the dimensional of the Hilbert space. The conjugate transpose of a state

is denoted as $\langle\psi|$ and it is convention to define the normalization of a state implicitly, so that the inner product of a state with itself is given by

$$\langle\psi|\psi\rangle = 1, \ \forall \ |\psi\rangle \ \in \ \mathcal{H}, \tag{1.1}$$

The dimension d of a quantum system is defined as the maximum number of states that can be completely distinguished from each other with a single measurement. Two distinguishable states $|\psi_i\rangle$, $|\psi_j\rangle$ are orthogonal, meaning $\langle\psi_i|\psi_j\rangle = \delta_{ij}$, where δ_{ij} is the Kronecker delta function. Consequently, a complete set of distinguishable states forms an orthonormal basis of the d-dimensional Hilbert space.

In many practical situations an observer may not have a complete understanding of the state of a quantum system. For example, an observer could know only that the system is in either state $|\psi\rangle$ with probability p_1, state $|\phi\rangle$ with probability p_2, and so on. In this case, the system is in a state described by a probabilistic ensemble of pure states. States of this kind are called mixed, and are represented mathematically by density operators, denoted ρ.

$$\rho = \sum_{i-1}^{n} p_i |\psi_i\rangle\langle\psi_i| \tag{1.2}$$

With this description, pure states are simply a special case of mixed states in which $n = 1$ and $p_1 = 1$. Density operators must satisfy a number of properties. Namely, the coefficients p_i of a density operator are all greater or equal to zero and must sum to one (since they represent probabilities). Density operators are always positive semi-definite (all eigenvalues are greater or equal to zero), denoted $\rho \geq 0$. Furthermore, every density operator is hermitian ($\rho = \rho^\dagger$, where $\dagger$ denotes the conjugate transpose) and must have trace equal to one ($\mathrm{Tr}[\rho] = 1$).

1.1.2 Measurements

Given an input quantum state ρ, all measurements in quantum mechanics can be represented as a box that outputs a classical variable i and a post-measurement quantum state σ_i. A Positive Operator Valued Measure (POVM) is a special type of quantum measurement in which the post-measurement state is discarded, and the measurement result is represented only by the classical variable i. An illustration of a POVM is shown in Figure 1.1. Single lines represent classical input/output and double lines represent quantum input/output.

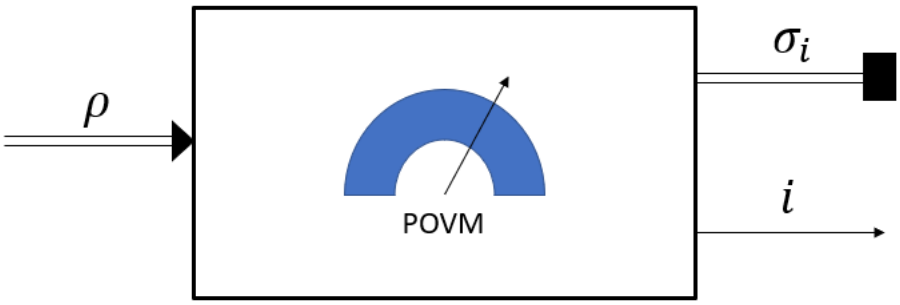

Figure 1.1: Graphical representation of a POVM measurement. The post measurement state σ_i is "traced out" and the classical variable i is recorded as the measurement outcome.

More formally, a POVM is a set of positive semi-definite operators $\{Q_i\}$ on the Hilbert space being measured that collectively satisfy the property

$$\sum_{i=0}^{n-1} Q_i = I, \tag{1.3}$$

where I is the identity matrix. With this notation, each Q_i is called an *effect* and corresponds to a single measurement outcome in an experiment. The probability of a particular outcome i occurring in an experiment is given by Born's rule

$$p_i = \mathrm{Tr}[\rho Q_i], \tag{1.4}$$

where p_i is the probability of outcome i occurring if the system is in state ρ. One special type of POVM occurs when the measurement effects are orthogonal and indempotent ($Q_i Q_j = \delta_{ij} Q_i$). POVMs of this kind are called *projective measurements* and each Q_i can be written as $Q_i = |\psi_i\rangle\langle\psi_i|$ for some orthonormal basis $\{|\psi_i\rangle\}$. In this case, Born's rule reduces to

$$p_i = \langle\psi_i|\rho|\psi_i\rangle, \tag{1.5}$$

which for pure states $\rho = |\phi\rangle\langle\phi|$, simplifies to

$$p_i = |\langle\phi|\psi_i\rangle|^2. \tag{1.6}$$

In direct analogy to states, projective measurement effects can be thought of as "pure", while non-projective measurement effects can be thought of as "mixed".

1.1.3 Operations

In the Schrodinger picture, quantum states evolve over time as they interact with their environment. This evolution is represented by a map over the Hilbert space. Deterministic quantum evolution is represented by a completely positive trace preserving (CPTP) map, also known as a *quantum channel* or quantum operation. These maps preserve the positivity of eigenvalues and the value of the trace of an input density operator and are commonly denoted $\mathcal{E}(\rho)$. In short, a quantum channel evolves a quantum state into another quantum state.

It is worth emphasizing that the only constraints on the definition of a quantum channel is that it is CPTP, which means that an output state may have an entirely different dimension than the input state. Furthermore, a quantum channel need not output a state in the same Hilbert space as the input state.

Unitary channels are a special subclass of channels that will be used throughout this thesis. These channels preserve purity in that they take pure quantum states to pure quantum states. Unitary channels are characterized by a unitary matrix U in the following way:

$$\mathcal{E}(\rho) = U\rho U^\dagger, \tag{1.7}$$

Where $U^\dagger U = UU^\dagger = I$. If instead of a density operator, one uses the pure state $|\psi\rangle$ as input, then the unitary channel can be written as $\mathcal{E}(|\psi\rangle) = U|\psi\rangle$.

1.2 Qubit Quantum Mechanics

1.2.1 State Space for Qubits

A quantum system described by a two-dimensional Hilbert space is known as a qubit. The general form of a qubit pure state is $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ where α and β are complex numbers that adhere to the constraint $|\alpha|^2 + |\beta|^2 = 1$, and the orthogonal states $|0\rangle$, $|1\rangle$ define the *computational basis* for qubits.

A 2×2 hermitian operator can be written as a linear combination of the 2×2 identity matrix as well as the three Pauli matrices:

$$\sigma_1 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \qquad \sigma_2 = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \qquad \sigma_3 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \tag{1.8}$$

The Pauli matrices are traceless and have eigenvalues ± 1. In addition, they satisfy $\mathrm{Tr}[\sigma_i \sigma_j] = 2\delta_{ij}\ \forall\ i, j$ and $\sigma_i^2 = I\ \forall\ i$. With this notation, the general density operator for a qubit can be written as

$$\rho = \frac{s_0}{2} I + \frac{1}{2} \sum_{i=1}^{3} s_i \sigma_i, \tag{1.9}$$

where $s_i = \mathrm{Tr}[\rho \sigma_i]$ are called the Bloch coefficients of the state. Noticing that $s_0 = \mathrm{Tr}[\rho I] = 1$, we define the Bloch vector as $\boldsymbol{s} = (1, s_1, s_2, s_3)$. For convenience, we will also separately define the vector $\tilde{\boldsymbol{s}} = (s_1, s_2, s_3)$, which excludes the contribution from the identity matrix. Using the fact that $\mathrm{Tr}[\rho^2] \leq 1$, and the properties of the Pauli matrices, we can derive

$$||\tilde{\boldsymbol{s}}|| = \sqrt{s_1^2 + s_2^2 + s_3^2} \leq 1, \tag{1.10}$$

with equality if and only if ρ is a pure state. From Eq. (1.10), we can infer that any qubit state is fully specified by the vector $\tilde{\boldsymbol{s}}$ with coordinates that lie within a sphere of radius 1 commonly called the *Bloch sphere*. Pure states lie on the surface of the Bloch sphere, while mixed states lie in the interior of the Bloch sphere. The Bloch sphere defines the state space for qubits, which we denote $\boldsymbol{\mathcal{S}}^{\mathrm{qubit}}$.

The condition $\mathrm{Tr}[\rho^2] \leq 1$ is informationally equivalent to the condition $\rho \geq 0$ for qubits. Therefore, there is no need to check explicitly that the eigenvalues of the density operator are positive. However, this fact does not hold for higher-dimensional systems. When deriving the state space for qutrit systems in section 1.3, the positivity of the density operator eigenvalues leads to additional constraints on the quantum state space.

1.2.2 Measurement Space for Qubits

The measurement space for qubits, denoted $\boldsymbol{\mathcal{E}}^{\mathrm{qubit}}$, is defined as the set of all 2×2 quantum mechanical effect matrices corresponding to an outcome in a POVM. Recall that any effect Q belonging to a POVM must satisfy $0 \leq Q \leq I$. In analogy to qubit states, effects can be written in their Bloch decomposition as:

$$Q = e_0 I + \sum_{i=1}^{3} e_i \sigma_i, \tag{1.11}$$

where $e_i = \frac{1}{2}\mathrm{Tr}[Q\sigma_i]$ are the Bloch coefficients of the effect and $\sigma_0 = I$. The trace of a qubit effect is given by $\mathrm{Tr}[Q] = 2e_0$, since the Pauli matrices are traceless. From $Q \leq I$, it

must hold that $\mathrm{Tr}[Q] \leq \mathrm{Tr}[I] = 2$. This implies $e_0 \leq 1$ and by using $Q \geq 0$, this condition extends to:

$$0 \leq e_0 \leq 1 \tag{1.12}$$

The effect in which $e_0 = 1$ is called the unit effect, which has the property $\mathrm{Tr}[Q\rho] = \mathrm{Tr}[I\rho] = 1$ for any density matrix ρ. One can show that the eigenvalues of an effect Q are:

$$x_i = e_0 \pm ||\tilde{\boldsymbol{e}}||, \tag{1.13}$$

where $\tilde{\boldsymbol{e}} = (e_1, e_2, e_3)$. Because $0 \leq Q \leq I$, the eigenvalues of Q must satisfy $0 \leq x_i \leq 1$. Therefore, using Eq. (1.13), the qubit measurement space is fully defined by:

$$\vec{e} \in \boldsymbol{\mathcal{E}}^{\text{qubit}} \text{ iff } \begin{cases} 0 \leq e_0 \leq 1 \\ ||\tilde{\boldsymbol{e}}|| \leq \mathrm{Min}\left(e_0,\ 1 - e_0\right) \end{cases} \tag{1.14}$$

Thus, any qubit measurement effect has a Bloch vector $\vec{e}$ that lies within the intersection of two 4-dimensional cones with spherical bases. This shape is commonly referred to as the *Bloch diamond* or Bloch double cone.

1.3 Qutrit Quantum Mechanics

This section loosely follows references [12, 28]. For more information on qutrit quantum theory, the interested reader is directed to these references. In analogy to a qubit, a qutrit quantum system is described by a three-dimensional Hilbert space. The general form of a qutrit pure state naturally extends from that of a qubit: $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle + \gamma|2\rangle$, where $|\alpha|^2 + |\beta|^2 + |\gamma|^2 = 1$. The computational basis for a qutrit consists of three orthogonal states, $|0\rangle$, $|1\rangle$ and $|2\rangle$.

Any 3×3 hermitian operator can be written as a linear combination of the 3×3 identity matrix and the eight Gell-Mann matrices [53]:

$$\lambda_1 = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad \lambda_2 = \begin{pmatrix} 0 & -i & 0 \\ i & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \qquad \lambda_3 = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

$$\lambda_4 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}, \qquad \lambda_5 = \begin{pmatrix} 0 & 0 & -i \\ 0 & 0 & 0 \\ i & 0 & 0 \end{pmatrix}, \qquad \lambda_6 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \tag{1.15}$$

$$\lambda_7 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -i \\ 0 & i & 0 \end{pmatrix}, \qquad \lambda_8 = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -2 \end{pmatrix}.$$

They share similar properties to the Pauli matrices in that $\mathrm{Tr}[\lambda_i] = 0 \; \forall \; i$ and $\mathrm{Tr}[\lambda_i \lambda_j] = 2\delta_{ij} \; \forall \; i, j$. Additionally, the Gell-Mann matrices satisfy the property $\mathrm{Tr}[\lambda_i \lambda_j \lambda_k] = 2g_{ijk} + if_{ijk} \; \forall \; i, j, k$. Here, g_{ijk} and f_{ijk} are elements of the three level structure tensors of group $SU(3)$. These tensors are derived from the commutation relations of the Gell-Mann matrices as follows:

$$\left[\frac{\lambda_i}{2}, \frac{\lambda_J}{2}\right] = i f_{ijk} \frac{\lambda_k}{2}, \tag{1.16}$$

$$\{\lambda_i, \lambda_J\} = \frac{4}{3}\delta_{ij} + 2g_{ijk}\lambda_k. \tag{1.17}$$

The tensor f is completely anti-symmetric and has non-zero elements (plus the anti-symmetric permutations) given by table 1.1. The tensor g is completely symmetric and has non-zero elements (plus symmetric permutations) given by table 1.2.

i, j, k	f_{ijk}
123	1
147	1/2
156	−1/2
246	1/2
257	1/2
345	1/2
367	−1/2
458	$\sqrt{3}/2$
678	$\sqrt{3}/2$

Table 1.1: Non-zero elements of the anti-symmetric structure tensor f.

i, j, k	g_{ijk}
118	$1/\sqrt{3}$
146	1/2
157	1/2
228	$1/\sqrt{3}$
247	−1/2
256	1/2
338	$1/\sqrt{3}$
344	1/2
355	1/2
366	−1/2
377	−1/2
448	$-1/(2\sqrt{3})$
558	$-1/(2\sqrt{3})$
668	$-1/(2\sqrt{3})$
778	$-1/(2\sqrt{3})$
888	$-1/\sqrt{3}$

Table 1.2: Non-zero elements of the completely symmetric structure tensor g.

1.3.1 State Space for Qutrits

Like its qubit counterpart, the density operator of a qutrit has a Bloch decomposition given by:

$$\rho = \frac{s_0}{3} I + \frac{1}{2} \sum_{i=1}^{8} s_i \lambda_i, \tag{1.18}$$

where the Gell-Mann matrices λ_i replace the Pauli matrices from the qubit case and $s_i = \mathrm{Tr}[\rho\lambda_i]$ are the Bloch coefficients of the qutrit state. Using $\mathrm{Tr}[\rho] = 1$, we see that the condition $s_0 = 1$ holds for qutrits. Furthermore, computing $\mathrm{Tr}[\rho^2]$ reveals:

$$\mathrm{Tr}[\rho^2] = \frac{1}{3} + \frac{1}{2}||\tilde{\boldsymbol{s}}||^2 \leq 1, \tag{1.19}$$

where $\tilde{\boldsymbol{s}} = (s_1, \ldots, s_8)$, and $||\tilde{\boldsymbol{s}}|| = \sqrt{s_1^2 + s_2^2 + \cdots + s_8^2}$. Equality holds if and only if ρ is a pure state. Eq. (1.19) can be rewritten as $||\tilde{\boldsymbol{s}}|| \leq \frac{2}{\sqrt{3}}$, which places the first constraint on the geometry of the qutrit state space $\boldsymbol{\mathcal{S}}^{\mathrm{qutrit}}$.

We now focus our attention to the condition $\rho \geq 0$. In contrast to the qubit case, the condition $\mathrm{Tr}[\rho^2] \leq 1$ is no longer informationally equivalent to the condition $\rho \geq 0$. Thus, ensuring that the density operator for a qutrit is positive-semidefinite is non-trivial. To do so, we begin by defining the *Bell polynomials* $B_k(x_1, x_2, \ldots, x_k)$ as follows:

$$B_k(x_1, x_2, \ldots, x_k) = \sum_{l=0}^{k} \binom{k}{l} x_{l+1} B_{k-l}(x_1, x_2, \ldots, x_{k-l}) \tag{1.20}$$

with $B_0 = 1$ [6]. The eigenvalues x_i of a matrix A are determined by solving its characteristic polynomial $p(x) = \det[A - xI] = 0$. The characteristic polynomial of an arbitrary $n \times n$ matrix A can be written in terms of the Bell polynomials as

$$p(x) = c_n x^n + c_{n-1} x^{n-1} + \cdots + (-1)^n \det[A] = 0, \tag{1.21}$$

where the coefficients c_{n-k} are given by:

$$c_{n-k} = \frac{(-1)^k}{k!} B_k(t_1,\ -t_2,\ 2t_3, \ldots,\ (-1)^{k-1}(k-1)! t_k), \tag{1.22}$$

and $t_i = \mathrm{Tr}[A^i]$ [56]. Using Eqs. (1.20), (1.21), and (1.22) with $n = 3$, we can write the characteristic polynomial for a qutrit density matrix as:

$$p_\rho(x) = c_3 x^3 + c_2 x^2 + c_1 x + c_0 = 0 \tag{1.23}$$

Where the coefficients are given explicitly by:

$$c_3 = B_0 = 1 \tag{1.24}$$
$$c_2 = -B_1(t_1) \tag{1.25}$$
$$c_1 = \frac{1}{2}B_2(t_1, -t_2) \tag{1.26}$$
$$c_0 = \frac{-1}{6}B_3(t_1, -t_2, 2t_3). \tag{1.27}$$

Pulling out the signs of the coefficients c_i, the characteristic polynomial can be rewritten as:

$$p_\rho(x) = x^3 - a_2x^2 + a_1x - a_0 = 0, \tag{1.28}$$

where $a_2 = -c_2$, $a_1 = c_1$, and $a_0 = -c_0$. To show that $\rho \geq 0$, the roots of Eq. (1.28) must all be positive (or zero). The positivity of the roots can be checked using the following theorem.

Theorem 1.1 (Decarte's Rule of Signs [3])
Let $p(x) = \sum_{k=0}^{n} c_{n-k}x^{n-k} = 0$. then the following must hold:

1. The number of positive roots of $p(x)$ is equal to the number of sign changes between consecutive coefficients, or less than that by an even amount.

2. The number of negative roots of $p(x)$ is equal to the number of sign changes between consecutive coefficients of $q(x) = p(-x)$, or less than that by an even amount.

By Decarte's rule of signs, the characteristic polynomial of a qutrit density operator in the form of Eq. (1.28) will have no negative roots if and only if all coefficients $a_i \geq 0$. In other words, $\rho \geq 0$ if and only if $a_i \geq 0 \ \forall \ i$. The coefficient $a_2 = B_1(t_1) = \mathrm{Tr}[\rho] = 1$ is positive by definition. Furthermore, $a_1 = \frac{1}{2}B_2(t_1, -t_2) = \frac{1}{2}(1 - \mathrm{Tr}[\rho^2]) \geq 0$ since $\mathrm{Tr}[\rho^2] \leq 1$. However, the coefficient a_0 is not always positive. The expanded form of a_0 is:

$$a_0 = \frac{1}{6}B_3(t_1, -t_2, 2t_3) = \frac{1}{6}(1 - 3\mathrm{Tr}[\rho^2] + 2\mathrm{Tr}[\rho^3]). \tag{1.29}$$

Consequently, the density operator of a qutrit state is positive semi-definite if and only if

$$1 - 3\mathrm{Tr}[\rho^2] + 2\mathrm{Tr}[\rho^3] = \frac{2}{9} - \frac{1}{2}||\tilde{\boldsymbol{s}}||^2 + \frac{1}{2}\sum_{i,j,k} g_{i,j,k}s_is_js_k \geq 0. \tag{1.30}$$

Because f is completely anti-symmetric, all terms involving this tensor cancel out in the derivation of Eq. (1.30), leaving only a dependence on the symmetric structure tensor g.

Equation (1.30) provides the final constraint on the qutrit state space, which can now be fully described by:

$$\boldsymbol{s} \in \boldsymbol{\mathcal{S}}^{\text{qutrit}} \text{ iff } \begin{cases} ||\tilde{\boldsymbol{s}}|| \leq \frac{2}{\sqrt{3}} \\ \frac{2}{9} - \frac{1}{2}||\tilde{\boldsymbol{s}}||^2 + \frac{1}{2}\sum_{i,j,k} g_{i,j,k} s_i s_j s_k \geq 0 \end{cases} \tag{1.31}$$

The first constraint in Eq. (1.310 describes an 8-dimensional hypersphere of radius $2/\sqrt{3}$, and is the qutrit analog of the well known Bloch sphere for qubits. The second constraint in Eq. (1.31) has no qubit counterpart and makes the geometry of the qutrit state space much more complicated.

1.3.2 Measurement Space for Qutrits

The measurement space $\boldsymbol{\mathcal{E}}^{\text{qutrit}}$ for qutrits is defined as the set of all valid quantum mechanical effects corresponding to an outcome in a POVM. Recall that any effect Q belonging to a POVM must satisfy $0 \leq Q \leq I$. Any qutrit POVM effect can be written in its Bloch decomposition as:

$$Q = e_0 I + \sum_{i=1}^{8} e_i \lambda_i, \tag{1.32}$$

where $e_0 = \frac{1}{3}\text{Tr}[Q]$, and $e_i = \frac{1}{2}\text{Tr}[Q\lambda_i]$ are the Bloch coefficients of the effect. From $Q \leq I$, it must hold that $\text{Tr}[Q] \leq \text{Tr}[I] = 3$. This implies $e_0 \leq 1$ and by using $Q \geq 0$, the condition extends to:

$$0 \leq e_0 \leq 1. \tag{1.33}$$

The effect in which $e_0 = 1$ is called the unit effect, which has the property $\text{Tr}[Q\rho] = \text{Tr}[I\rho] = 1$ for any density matrix ρ.

In analogy to the analysis for qutrit states, there are additional constraints that arise from the property $Q \geq 0$. Using Eq. (1.28), we can find conditions on the effect Bloch vector using Decarte's rule of signs. For quantum effects, the coefficients a_i are given explicitly as:

$$a_2 = B_1(t_1) = \text{Tr}[Q] = 3e_0 \tag{1.34}$$

$$a_1 = \frac{1}{2}B_2(t_1, -t_2) = \frac{1}{2}(9e_0^2 - \text{Tr}[Q^2]) \tag{1.35}$$

$$a_0 = \frac{1}{6}B_3(t_1, -t_2, 2t_3) = \frac{1}{6}(27e_0^3 - 9e_0\text{Tr}[Q^2] + 2\text{Tr}[Q^3]). \tag{1.36}$$

As long as the coefficients $a_i \geq 0$, then Q is positive semi-definite. Using Eq. (1.33), $a_2 \geq 0$ by definition. The coefficient $a_1 \geq 0$ if and only if

$$||\tilde{\boldsymbol{e}}|| \leq \sqrt{3} e_0, \tag{1.37}$$

where $\tilde{\boldsymbol{e}} = (e_1, \ldots, e_8)$, and $||\tilde{\boldsymbol{e}}|| = \sqrt{e_1^2 + \cdots + e_8^2}$. Finally, by following the same analysis that resulted in Eq. (1.30), the coefficient $a_0 \geq 0$ if and only if:

$$e_0^3 - e_0 ||\tilde{\boldsymbol{e}}||^2 + \frac{2}{3} \sum_{i,j,k} g_{i,j,k} e_i e_j e_k \geq 0. \tag{1.38}$$

Eq. (1.38) provides another constraint on the geometry of the qutrit effect space $\boldsymbol{\mathcal{E}}^{\text{qutrit}}$. There are two remaining constraints that are found by examining the other effects within the POVM. Specifically, for any POVM, it must hold that $Q_0 + \sum_{i=1}^{n-1} Q_i = I$. By definition, $\sum_i Q_i = \bar{Q}_0 = I - Q_0$. This implies that $\bar{Q}_0$ is simply Q_0 under the mapping $e_0 \to 1 - e_0$ and $e_i \to -e_i$. Therefore, two more constraints on the effect Bloch vector can be identified by applying this mapping to Eqs. (1.37) and (1.38). The qutrit effect space can be fully described by:

$$\boldsymbol{e} \in \boldsymbol{\mathcal{E}}^{\text{qutrit}} \text{ iff } \begin{cases} 0 \leq e_0 \leq 1 \\ ||\tilde{\boldsymbol{e}}|| \leq \sqrt{3} \text{Min}\left(e_0,\ 1 - e_0\right) \\ e_0^3 - e_0 ||\tilde{\boldsymbol{e}}||^2 + \frac{2}{3} \sum_{i,j,k} g_{ijk} e_i e_j e_k \geq 0 \\ (1 - e_0)^3 - (1 - e_0) ||\tilde{\boldsymbol{e}}||^2 - \quad \dfrac{2}{3} \displaystyle\sum_{i,j,k} g_{ijk} e_i e_j e_k \geq 0. \end{cases} \tag{1.39}$$

1.3.3 Geometry of Qutrit Quantum Mechanics

Since the qutrit state space is eight-dimensional and the qutrit effect space is nine-dimensional, their geometric featured cannot be viewed directly. Instead, we visulaize the geometry of the qutrit state and effect spaces by computing various three-dimensional projections of the full-dimensional spaces. To do so, we make use of the *joint numerical range* of a set of k operators $\{A_i\}_{i=1}^k$, defined as [7]

$$W(A_0, \ldots, A_{k-1}) = \{\boldsymbol{w} \in \mathbb{R}^k : w_\alpha = \langle \psi | A_\alpha | \psi \rangle, \forall |\psi\rangle \in \mathbb{C}^d\}, \tag{1.40}$$

where $\mathbb{C}^d$ is the complex vector space of dimension d. For the set of nine Gell-Mann matrices (including the identity matrix), which forms an orthonormal basis of hermitian

traceless operators, the joint numerical range $W(\lambda_0, \ldots, \lambda_8)$ specifies the set of all pure state vectors $\boldsymbol{s}$ in the qutrit state space, where $\lambda_0 = I$. That is,

$$\boldsymbol{\mathcal{S}}^{\text{qutrit}}_{\text{pure}} = W(\lambda_0, \ldots, \lambda_8). \tag{1.41}$$

It follows that the *convex hull* of $W(\lambda_0, \ldots, \lambda_8)$ is equal to the set of 9-dimensional Bloch vectors characterized by Eq. (1.31). That is, the set of all qutrit states (pure and mixed),

$$\boldsymbol{\mathcal{S}}^{\text{qutrit}} = \{\boldsymbol{s} \in \mathbb{R}^9 : s_\alpha = \text{Tr}[\rho\lambda_\alpha],\ \forall \rho \in \mathcal{D}(\mathbb{C}_3)\}, = \text{cvxhull}\left[W(\lambda_0, \ldots, \lambda_8)\right],$$

where $\mathcal{D}(\mathbb{C}_3)$ is the set of all 3×3 density operators. For any three of these nine operators, $\{\lambda_\mu, \lambda_\nu, \lambda_\omega\}$, the convex hull of their joint numerical range describes the *projection* of $\boldsymbol{\mathcal{S}}^{\text{qutrit}}$ into the 3-dimensional operator subspace spanned by $\{\lambda_\mu, \lambda_\nu, \lambda_\omega\}$,

$$P_{\mu\nu\omega}(\boldsymbol{\mathcal{S}}^{\text{qutrit}}) = \{\boldsymbol{s} \in \mathbb{R}^3 : s_\alpha = \text{Tr}[\rho\lambda_\alpha], \forall \rho \in \mathcal{D}(\mathbb{C}_3)\} = \text{cvxhull}\left[W(\lambda_\mu, \lambda_\nu, \lambda_\omega)\right]. \tag{1.42}$$

Similar considerations hold for the effects. Taking

$$\tilde{\lambda}_0 = \frac{1}{3}I, \quad \tilde{\lambda}_\alpha = \frac{1}{2}\lambda_\alpha,\ \forall \alpha \in \{1, \ldots, 8\},$$

the joint numerical range $W(\tilde{\lambda}_0, \ldots, \tilde{\lambda}_8)$ is equal to the set of generalized Bloch vectors associated to *atomic* effects on qutrits. An atomic quantum effect is one described by a rank-1 projector $\{|\psi\rangle\langle\psi| : |\psi\rangle \in \mathbb{C}^3\}$. Denoting the set of effect vectors that are associated to atomic qutrit effects by $\boldsymbol{\mathcal{E}}^{\text{qutrit}}_{\text{atomic}}$, we have

$$\boldsymbol{\mathcal{E}}^{\text{qutrit}}_{\text{atomic}} = W(\tilde{\lambda}_0, \ldots, \tilde{\lambda}_8). \tag{1.43}$$

For all atomic qutrit effect vectors, $e_0 = \text{Tr}(\frac{1}{3}|\psi\rangle\langle\psi|) = \frac{1}{3}$.

To extend our analysis from the set of atomic quantum effect vectors to the set of *all* quantum effect vectors, we use the following procedure. For every effect Q in $\boldsymbol{\mathcal{E}}^{\text{qutrit}}_{\text{atomic}}$, we add to the set its complement, $I - Q$. Additionally, we add to the set the unit effect, I (with effect vector $\boldsymbol{u} = (1, 0, \ldots, 0)$), and the zero effect, 0 (with effect vector $\boldsymbol{0} = (0, \ldots, 0)$), which assigns zero probability when paired with all states. Finally, we compute the convex hull of the resulting set. In summary, if one denotes the set of complements of atomic quantum effects by $\bar{\boldsymbol{\mathcal{E}}}^{\text{qutrit}}_{\text{atomic}}$, then the full set of qutrit quantum effect vectors satisfying Eq. (1.39) is given by

$$\begin{aligned}\boldsymbol{\mathcal{E}}^{\text{qutrit}} &= \{\boldsymbol{e} \in \mathbb{R}^9 : e_\alpha = \text{Tr}[Q\tilde{\lambda}_\alpha],\ \forall Q \in \text{QEff}(\mathbb{C}^3)\} \\ &= \text{cvxhull}\left[\{\boldsymbol{0}\} \cup \{\mathbf{u}\} \cup \boldsymbol{\mathcal{E}}^{\text{qutrit}}_{\text{atomic}} \cup \bar{\boldsymbol{\mathcal{E}}}^{\text{qutrit}}_{\text{atomic}}\right],\end{aligned} \tag{1.44}$$

where $\mathrm{QEff}(\mathbb{C}^3)$ is the set of all 3×3 positive semi-definite operators with eigenvalues less than or equal to 1.

Following the same steps as for qutrit states, we wish to consider the *projection* of $\boldsymbol{\mathcal{E}}^{\mathrm{qutrit}}$ into the 3-dimensional operator subspace spanned by any three operators $\{\tilde{\lambda}_\mu, \tilde{\lambda}_\nu, \tilde{\lambda}_\omega\}$. Let $\mathbf{0}^{(3)} = (0,0,0)$ be the 3-dimensional projection of $\mathbf{0}$ for any choice of μ, ν, ω. Let $\mathbf{u}^{(\mu\nu\omega)}$ denote the projection of $\mathbf{u}$ into a 3-dimensional subspace specified by the choice of μ, ν, ω. If $0 \in \{\mu, \nu, \omega\}$, then $\mathbf{u}^{(\mu\nu\omega)} = (1,0,0)$, while if $0 \notin \{\mu, \nu, \omega\}$, then $\mathbf{u}^{(\mu\nu\omega)} = (0,0,0)$. Finally, let $\bar{W}(B_\mu, B_\nu, B_\omega)$ denote the complement of $W(\tilde{\lambda}_\mu, \tilde{\lambda}_\nu, \tilde{\lambda}_\omega)$ relative to $\mathbf{u}^{(\mu\nu\omega)}$, i.e., $\bar{W}(\tilde{\lambda}_\mu, \tilde{\lambda}_\nu, \tilde{\lambda}_\omega) = \{\mathbf{u}^{(\mu\nu\omega)} - \mathbf{e} : \mathbf{e} \in W(\tilde{\lambda}_\mu, \tilde{\lambda}_\nu, \tilde{\lambda}_\omega)\}$. For those projections where $\mathbf{u}^{(\mu\nu\omega)} = (0,0,0)$, $\bar{W}(\tilde{\lambda}_\mu, \tilde{\lambda}_\nu, \tilde{\lambda}_\omega)$ is simply the inversion about the origin of $W(\tilde{\lambda}_\mu, \tilde{\lambda}_\nu, \tilde{\lambda}_\omega)$. Relative to these definitions, the 3-dimensional projections of $\boldsymbol{\mathcal{E}}^{\mathrm{qutrit}}$ are given by:

$$\begin{aligned} P_{\mu\nu\omega}(\boldsymbol{\mathcal{E}}^{\mathrm{qutrit}}) &= \{\boldsymbol{e} \in \mathbb{R}^3 : e_\alpha = \mathrm{Tr}[Q\tilde{\lambda}_\alpha], \forall Q \in \mathrm{QEff}(\mathbb{C}^3)\} \\ &= \mathrm{cvxhull}\big[\{\mathbf{0}^{(3)}\} \cup \{\mathbf{u}^{(\mu\nu\omega)}\} \cup W(\tilde{\lambda}_\mu, \tilde{\lambda}_\nu, \tilde{\lambda}_\omega) \cup \bar{W}(\tilde{\lambda}_\mu, \tilde{\lambda}_\nu, \tilde{\lambda}_\omega)\big]. \end{aligned} \tag{1.45}$$

Fig. 1.2 shows a number of 3-dimensional projections of the qutrit state space (dispayed as yellow shapes) and the qutrit effect space (displayed as cyan shapes). Before describing some of their features, we briefly explain how these plots are generated. To plot a given 3-dimensional projection of $\boldsymbol{\mathcal{S}}^{\mathrm{qutrit}}$, we begin by generating a large quantity of random GPT state vectors satisfying Eq. (1.31). For every GPT state vector in our random set, we project it down into the 3-dimensional space of interest. We then plot the convex hull of these 3-dimensional vectors. Similarly, to generate a plot of any given 3-dimensional projection of $\boldsymbol{\mathcal{E}}^{\mathrm{qutrit}}$, we generate a large quantity of random GPT effect vectors corresponding to atomic qutrit effects (that is, satisfying Eq. (1.39) with $e_0 = \frac{1}{3}$). For every GPT effect vector obtained in this way, we compute its complement effect. We then project both the atomic effects and their complements down into the 3-dimensional space of interest. Finally, we plot the convex hull of the set of GPT effect vectors, their complements, and the projected zero and unit effect vectors.

As the number of vectors defining these convex shapes is always finite, the shapes depicted in the plots are polytopes. By close inspection of Fig. 1.2, one can see that each shape has some roughness compared to the true 3-dimensional projections that could be obtained from infinite sampling. Nonetheless, since the sampling is quite dense, the plots provide a close approximation to the true shapes.

We now discuss the 3-dimensional projections of the 9-dimensional qutrit state space onto three of the Gell-Mann matrices (i.e., not including the identity operator). There are $\binom{8}{3}$ =56 of these to choose from. In Ref. [57], it was shown that by utilizing the connection

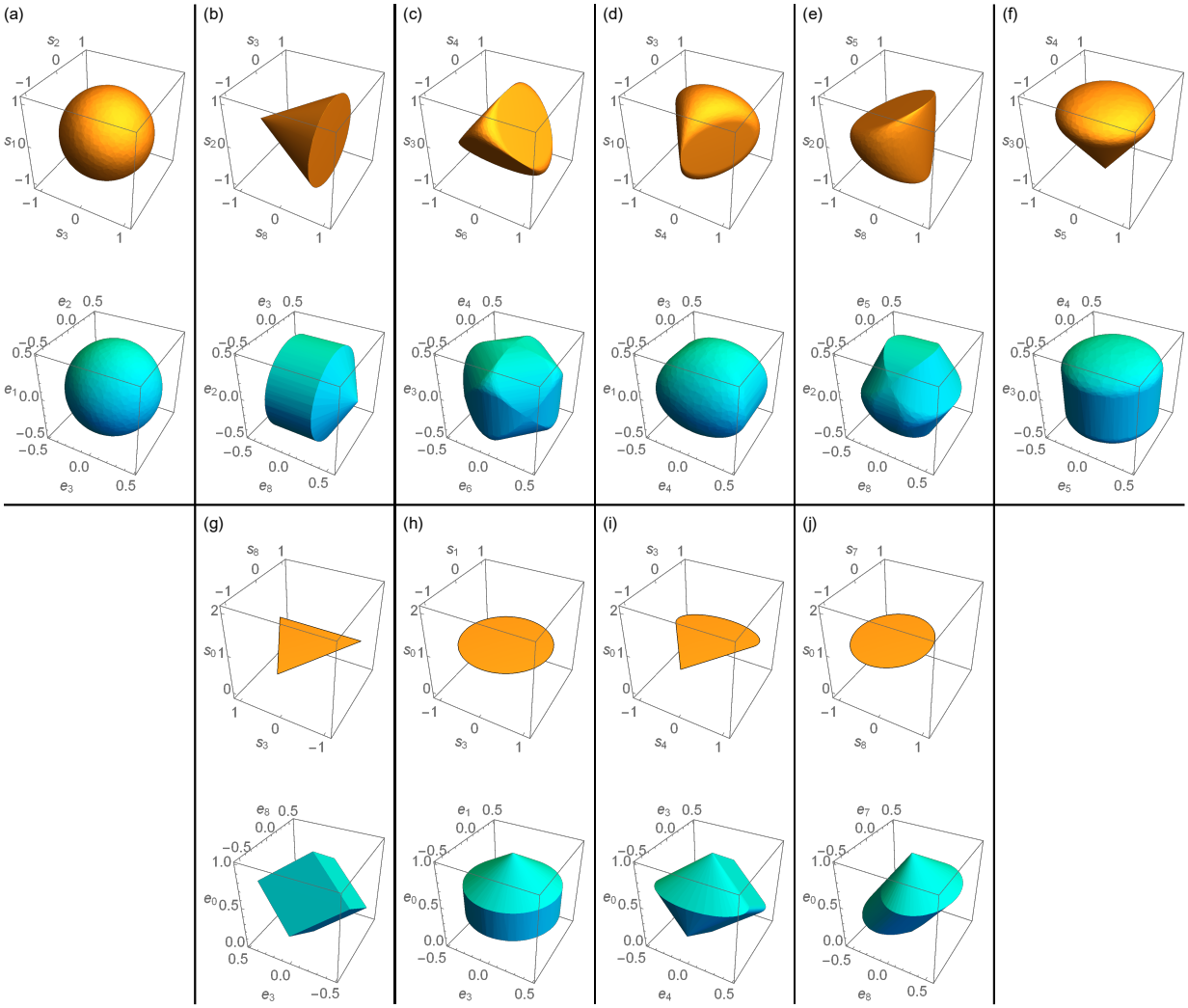

Figure 1.2: Three-dimensional projections of the 8-dimensional qutrit state space (yellow polytopes) and the 9-dimensional qutrit effect space (cyan polytopes). (a) $l = 0$, $f = 0$ projection: (s_1, s_2, s_3), (e_1, e_2, e_3). (b) $l = \infty$, $f = 1$ projection: (s_2, s_3, s_8), (e_2, e_3, e_8). (c) $l = 0$, $f = 4$ projection: (s_3, s_4, s_6), (e_3, e_4, e_6). (d) $l = 0$, $f = 2$ projection: (s_1, s_3, s_4), (e_1, e_3, e_4). (e) $l = 1$, $f = 2$ projection: (s_2, s_5, s_8), (e_2, e_5, e_8). (f) $l = \infty$, $f = 0$ projection: (s_3, s_4, s_5), (e_3, e_4, e_5). (g)-(j) Projections involving the e_0 Bloch coefficient. (g) $l = 3$ projection: $(1, s_3, s_8)$, (e_0, e_3, e_8). (h) $l = 0$ projection: $(1, s_1, s_3)$, (e_0, e_1, e_3). (i) $l = 2$ projection: $(1, s_3, s_4)$, (e_0, e_3, e_4). (j) Another $l = 0$ projection: $(1, s_7, s_8)$, (e_0, e_7, e_8).

to joint numerical ranges, these shapes are usefully classified into different types according to the number of *flat edges* (segments) in their boundary and the number of *flat faces* in their boundary. Thus, we specify each projection by the three components of $\boldsymbol{s}$ used to generate it, and then classify the type of the projection based on the number of flat edges l, and the number of flat faces f. For example, the sphere projection in Fig. 1.2(a) is characterized by the tuple $(P_{\mu\nu\omega}(\mathcal{S}^{\text{qutrit}}), l, f) = ((s_1, s_2, s_3), 0, 0)$, as it is obtained from the s_1, s_2, s_3 components of all $\boldsymbol{s} \in \boldsymbol{\mathcal{S}}^{\text{qutrit}}$, it has 0 flat edges, and it has 0 flat faces. Each projection in Fig. 1.2(a)-(f) shows a distinct type of 3-dimensional projection of the qutrit state space.

In Fig. 1.2(g)-(j), we display 3-dimensional projections of the qutrit state space where one of the axes (taken to be the vertical axis) corresponds to the s_0 component. As discussed earlier, this component has value 1 for all quantum states. Consequently, these 3-dimensional projections of the qutrit state space are only nontrivial along the other two axes. The resulting 2-dimensional projections can be classified in an analogous way to the 3-dimensional projections, namely, the number of flat edges in the shape. For example, Fig. 1.2(g) has $l = 3$ flat edges. Notice that Fig: 1.2(h) and (j) are of the same class in that they both have $l = 0$ flat edges. These two projections provide an example of a situation where two projections of the same type can have distinct visual representations. Although we do not present a full list of all possible projection types generated by any three of the nine Gell-Mann matrices in Fig. 1.2, the sample projections chosen are sufficient to give a sense of the possible geometric features that are characteristic to qutrit quantum mechanics.

We now turn to the projections of $\boldsymbol{\mathcal{E}}^{\text{qutrit}}$. In Fig. 1.2, we display projections of $\boldsymbol{\mathcal{E}}^{\text{qutrit}}$ alongside the matching projections of $\boldsymbol{\mathcal{S}}^{\text{qutrit}}$. In Fig. 1.2(g)-(j), we display a sampling of effect space projections that contain the e_0 component. Because the component e_0 associated to the identity matrix is nontrivial for effects, these projections are nontrivial along all three axes. The set of trace-one quantum effects is always precisely the same as the set of normalized quantum states. Consequently, the $e_0 = 1/3$ section of the space of GPT effect vectors for a qutrit should be the same shape as the $s_0 = 1$ section of the space of GPT states for a qutrit. This is exactly what is seen in Figs. 1.2(g)-(j). In the 3-dimensional projections where $\mathbf{u}^{(\mu\nu\omega)} = (0, 0, 0)$, we expect the projection of the effect space to be symmetric under inversion about the origin. This behaviour is observed in Figs. 1.2(a)-(f).

1.4 The Haar Measure

For a deeper understanding of the Haar-Measure and its application to quantum mechanics, [42] is a fantastic reference for further reading. In mathematics, a *measure* provides information for how some measurable parameter (such as length, area or volume) is distributed in a space. For example, when working in spherical coordinates, the measure $\rho \sin(\theta)$ dictates how the volume of the sphere is distributed in the space. The measure weights the volumes of different parts of the sphere differently based on where they are (close to the equator or close to the poles).

More generally, the measure that assigns an invariant volume to a particular space (group) is called the Haar measure and is formally defined as follows:

Definition 1.4.1. (Haar Measure [16]): If G is a compact topological group, the measure $\mu(G)$ is a normalized Haar measure if

- $\mu(G) = 1$.
- if S is some non-empty open subset of G, then $\mu(S) > 0$.
- if S is some non-empty open subset of G, then for $y \in S$ and $x \in G$, $\mu(y) = \mu(xy) = \mu(yx)$.

The last property dictates that the Haar measure is invariant under left or right multiplication by one group element on another. For example, suppose $G = SU(n)$ (the special unitary group of dimension n). Then the Haar measure on $SU(n)$ is invariant under the left or right action of an $n \times n$ unitary matrix U. The utility of the Haar measure is that it can be used to sample points uniformly from a space. We use this fact in the following section to generate quantum states that are uniformly distributed on the surface of the n-dimensional Bloch space. We call such a set of quantum states *Haar-distributed*.

1.4.1 Generating Haar-Distributed Quantum States

We will now derive an $SU(n)$ Haar invariant probability distribution and use it to generate a number of pure quantum states that are uniformly distributed throughout n-dimensional Bloch space. For this derivation, we closely follow reference [35].

Let X be a $n \times n$ matrix filled with complex, independent and identically distributed (i.i.d.) standard normal random variables. That is, $x_{ij} = a_{ij} + ib_{ij}$, where $a_{ij}, b_{ij} =$

Normal$(0,1)$. Then the joint probability density of the random variables comprising X is given by

$$\mathrm{pr}(X) = \frac{1}{(2\pi)^{\frac{n^2}{2}}} \prod_{i,j=1}^{n} e^{\frac{-x_{ij}^2}{2}} = \frac{1}{(2\pi)^{\frac{n^2}{2}}} e^{-\frac{1}{2}\sum_{i,j=1}^{n} x_{ij}^2}. \tag{1.46}$$

In order to prove that the probability density $\mathrm{pr}(X)$ is $SU(n)$ Haar-invariant, we need to show that the action of a unitary matrix U on X does not change its probability density. Let $Y = UX$ so that $\prod_{i,j=1}^{n} dy_{ij} = |\det(U^{-1})| \prod_{i,j=1}^{n} dx_{ij}$. Then by substitution, the joint probability density of the matrix UX with respect to Y becomes

$$\frac{|\det(U^{-1})|}{(2\pi)^{\frac{n^2}{2}}} e^{-\frac{1}{2}\sum_{i,j=1}^{n}[U^{-1}Y]_{ij}^2} = \frac{1}{(2\pi)^{\frac{n^2}{2}}} e^{-\frac{1}{2}\sum_{i,j=1}^{n} y_{ij}^2}, \tag{1.47}$$

where we used the fact that the determinant of a unitary matrix in $SU(n)$ is 1, and that unitary matrices preserve length, $\sum_{i,j=1}^{n}[U^{-1}Y]_{ij}^2 = ||U^{-1}Y||^2 = ||Y||^2 = \sum_{i,j=1}^{n} y_{ij}^2$. Thus, we have shown that the group action of the matrix U does not change the probability density of the matrix X and so $\mathrm{pr}(X)$ is said to be an $SU(n)$ invariant Haar distribution.

Given a random matrix X with probability distribution governed by equation (1.46), we can derive a Haar-distributed unitary matrix by taking its QR decomposition, $X = QR$, where Q is a unitary matrix and R is an upper right triangular matrix. For more details on the QR decomposition, see Ref. [11]. Special care must be taken as the decomposition $X = QR$ is not unique. In fact, for any invertible matrix Λ, $X = Q\Lambda\Lambda^{-1}R$. In reference [36], the authors show that the unitary matrix $Q\Lambda$ will be Haar-distributed if and only if the diagonal elements of $\Lambda^{-1}R$ are real and strictly positive. Choosing $\Lambda = \mathrm{diag}(\frac{R_{11}}{|R_{11}|}, \frac{R_{22}}{|R_{22}|}, \ldots, \frac{R_{nn}}{|R_{nn}|})$ forces this to be the case. Thus, the matrix $U = Q\Lambda$ is guaranteed to be a Haar-distributed unitary matrix.

The columns and rows of a unitary matrix are orthonormal. Therefore, it must be true that any row or column of an $n \times n$ unitary matrix must correspond to a pure n-dimensional quantum state. Taking the first column (or row) of the Haar distributed unitary matrix U derived from the QR decomposition of X yields an n-dimensional Haar-distributed quantum state. A set of such states would be uniformly distributed over the n-dimensional Bloch space.

To summarize, we can generate a set of N Haar-Distibuted n-dimensional pure quantum states as follows:

1. Generate N complex $n \times n$ matrices $\{X_i\}_{i=1}^{N}$ with i.i.d. standard normal elements.
2. Perform the QR decomposition on each X_i to obtain a set of unitary matrices $\{Q_i\}_{i=1}^{N}$.

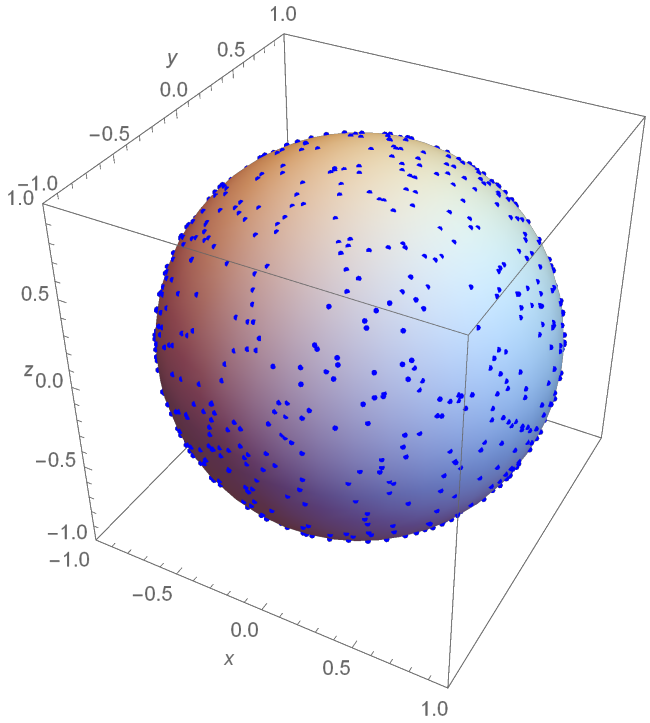

Figure 1.3: 1000 Haar-Distributed qubit states on the Bloch sphere.

3. Compute $U_i = Q_i\Lambda_i$, where $\Lambda_i = \text{diag}(\frac{R_{i,11}}{|R_{i,11}|}, \frac{R_{i,22}}{|R_{i,22}|}, \ldots, \frac{R_{i,nn}}{|R_{i,nn}|})$ to obtain a set $\{U_i\}_{i=1}^N$ of Haar-distributed unitary matrices.

4. Take the first column of each U_i to construct a set $\{|\psi_i\rangle_i\}_{i=1}^N$ of pure n-dimensional Haar-distributed quantum states.

Figure 1.3 shows 1000 Haar-distributed states on the Bloch sphere that were generated using the above algorithm. This algorithm was also used in section 1.3.3 to sample a large number of Haar-distributed qutrit quantum states and effects. These sets of states and effects were used to plot the various 3-dimensional projections shown in figure 1.2. We will make use of this algorithm again in chapter 4 for generating the preparations and measurements used in our experiment. The Mathematica code used to implement this algorithm can be found in appendix A.1.

Chapter 2

Experimental Quantum Optics

The data obtained in this thesis was obtained through quantum optical experiments. This chapter will outline the necessary concepts the reader requires and is written with an emphasis on polarization and interference. The chapter utilizes a mixture of concepts from [24, 32, 29]. The reader is directed to these references to obtain a more thorough understanding of the content of this chapter.

There are a variety of systems that can be utilized to encode quantum states. In fact, photons, trapped ions, superconducting systems, and systems involving nuclear magnetic resonance of spin states can all be used for this purpose. There is no single implementation that claims superiority over the others in all situations. Instead, each encoding method is best suited for specific applications. In this thesis, we work with a photonic system, which is useful because the fundamental elements (photons) interact very little with each other or with the environment. This is advantageous as it facilitates the ability for coherence to be maintained for long periods of time. Photonic systems are stable at room temperature, and single photons can be easily generated using spontaneous parametric down conversion (SPDC). Additionally, operations on photons can be implemented using only linear optical components such as beam splitters and linear retarders (wave plates), making it relatively easy to prepare and measure quantum states.

2.1 Photonic Quantum States

A photon is defined as an excitation of the electromagnetic field. Although the quantum mechanical state space of a single photon is infinite-dimensional, the polarization degree of

freedom of a photon is only two-dimensional. Thus, a single qubit state can be constructed from only the polarization degree of freedom. As long as we can ensure that the photons we use in our experiment are nearly identical in every way except polarization, we can fully describe the quantum state of a photon based solely on its polarization. For photonic qubits, we take the computational basis to be the horizontal ($|H\rangle$) and vertical ($|V\rangle$) polarizations. These polarizations are orthogonal, and so an arbitrary pure qubit state can be written as:

$$|\psi\rangle = \cos(\theta)|H\rangle + e^{i\phi}\sin(\theta)|V\rangle, \tag{2.1}$$

where θ and ϕ represent the polar and azimuthal angles specifying the location of the state on the Bloch sphere defined in section 1.2.1.

There are two other commonly used bases for photonic qubits. The first is the diagonally polarized basis $|D\rangle/|A\rangle$, which when written in terms of the computational basis, has states defined as: $|D\rangle = \frac{1}{\sqrt{2}}(|H\rangle + |V\rangle)$ and $|A\rangle = \frac{1}{\sqrt{2}}(|H\rangle - |V\rangle)$. The other commonly used basis for photonic qubits is the circularly polarized basis $|R\rangle/|L\rangle$, which when written in terms of the computational basis, has states defined as: $|R\rangle = \frac{1}{\sqrt{2}}(|H\rangle + i|V\rangle)$ and $|A\rangle = \frac{1}{\sqrt{2}}(|H\rangle - i|V\rangle)$. Qubit states with real coefficients are said to be linearly polarized and qubit states with complex coefficients are said to be elliptically polarized.

We make use of the polarization and spatial degrees of freedom of photons to generate three-level states in our experiment.

2.2 Wave Plates

An arbitrary single qubit unitary operation can be performed on the polarization degree of freedom of a photon using three pieces of birefringent glass by varying the angles of their respective optical axes. Birefringent materials have two indices of refraction along two independent axes called the *fast* and *slow* axes. Light travelling along the slow axis of the material will be delayed and acquire a relative phase delay compared to light travelling along the fast axis. That is, the birefringent material implements the map

$$\begin{aligned} |f\rangle &\rightarrow |f\rangle \\ |s\rangle &\rightarrow e^{i\phi}|s\rangle \end{aligned} \tag{2.2}$$

where $|f\rangle$ and $|s\rangle$ are the photon polarization states aligned to the fast and slow axes, respectively.

A half wave plate (HWP) is an example of a birefringent piece of glass in which the relative phase acquired by $|s\rangle$ is π, while a quarter wave plate (QWP) is a piece of birefringent glass in which the relative phase acquired by $|s\rangle$ is $\pi/2$. In most cases, the only physical difference between a half and quarter wave plate is its thickness (increased thickness = increased phase delay). The action of a wave plate on a polarization state can be changed by rotating its fast axis (optical axis) relative to the horizontal. The unitary operators which represent the action of a HWP and QWP on a polarization state are given by

$$\mathrm{HWP}[\theta] = \begin{pmatrix} \cos(2\theta) & \sin(2\theta) \\ \sin(2\theta) & \cos(2\theta) \end{pmatrix} \tag{2.3}$$

$$\mathrm{QWP}[\theta] = \begin{pmatrix} \cos^2(\theta) - i\sin^2(\theta) & (1+i)\cos(\theta)\sin(\theta) \\ (1+i)\cos(\theta)\sin(\theta) & -i\cos^2(\theta) + \sin^2(\theta) \end{pmatrix}, \tag{2.4}$$

where θ is the angle between the fast axis and the horizontal axis.

A HWP or a QWP provides a fixed relative delay. In some cases, it is necessary to have a delay that is variable from 0 to 2π. One easy way to accomplish a variable delay is to rotate a piece of birefringent material (such as a wave plate) about the axis perpendicular to the beam. This effectively increases the amount of glass that a photon travels through, and consequently increases the relative delay. Variable retarders of this kind are used in our experiment to control the relative time delay between photons travelling along separate arms of an interferometer in an attempt to control interference visibility. We discuss interference visibility in section 2.4.

2.3 Qubit Operations and Measurements

The detectors used in the experiment are not polarization sensitive. To measure a photon's polarization state, we use polarizing beam splitters (PBSs) to separate its horizontally polarized and vertically polarized components into different spatial paths. A PBS reflects vertically polarized light and transmits horizontally polarized light. Detectors are placed along the two spatial paths to measure in the computational basis.

To perform an arbitrary unitary operation on a qubit polarization state, a so called QWP-HWP-QWP device must be employed. However in many experiments, we only require the ability to convert from one arbitrary polarization state $|\psi\rangle$ to the horizontally polarized state $|H\rangle$. To do this, only a HWP-QWP device (or QWP-HWP device) is required. Using a HWP-QWP device followed by a PBS as shown in figure 2.1, we can

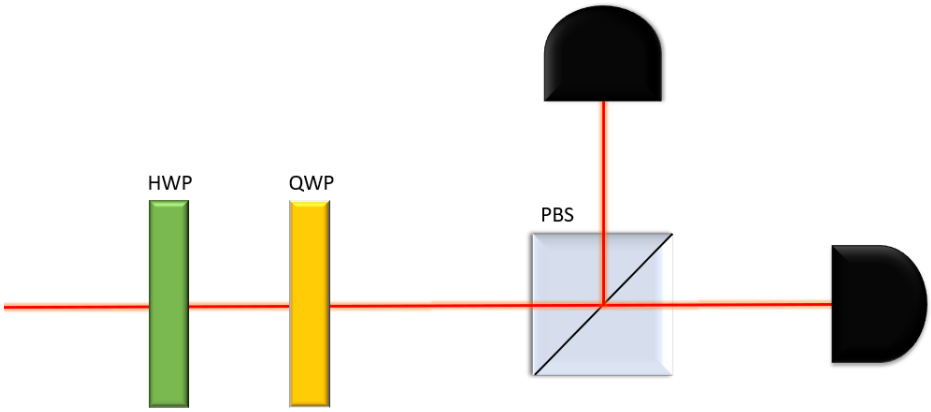

Figure 2.1: Example of a polarization analyzer using a HWP-QWP device and a PBS. A projective measurement can be performed on any arbitrary qubit polarization state by adjusting the angles of the half and quarter wave plates.

compute the probability of measuring the state $|\psi\rangle$ in the transmitted port of the PBS using Born's rule:

$$pr_H = |\langle H|QWP[\phi]HWP[\theta]|\psi\rangle|^2. \tag{2.5}$$

To find the wave plate angles necessary to measure $|\psi\rangle$ in the transmitted port of the PBS with unit probability, we numerically maximize over wave plate angles θ and ϕ in equation 2.5. We use this principle when computing the necessary wave plate angles needed to prepare and measure the three-level system in our experiment. We discuss this technique in section 4.2.4.

2.4 Coherence and Interference

For a detailed discussion of interference and coherence, the reader is directed to Ref. [31].

Two wave sources are said to be coherent if their frequencies and waveforms are identical. Interference between two beams of light can only occur if they are spatially overlapped and if some degree of coherence exists between them. Any real light source is made up of an ensemble of individual atoms which each radiate light with well defined statistical properties. Collisions with other atoms, as well as differences in the velocities of the individual emitters causes a range of frequencies to be present in the output of the source.

As a result, light emitted from a source at one moment in time may differ significantly in phase compared to light emitted at a later time. As the time difference between two waves emitted from the same source increases, the coherence of of the two waves degrades. The characteristic time that it takes for the coherence to drop to zero is called the *coherence time*, denoted τ_c. The coherence time is approximated as the inverse of the spectral bandwidth.

The coherence time of a source can be measured using an interferometer. For example, consider the Mach-Zehnder interferometer shown in Fig. 2.2. Laser light is spatially separated onto different paths using a 50:50 beam splitter. Light travelling along the upper path is variably delayed by time τ relative to light travelling along the bottom path. The two beams interfere once recombined on a second 50:50 beam splitter. When the delay is set such that $\tau > \tau_c$, the interference disappears as the two beams are no longer coherent with each other.

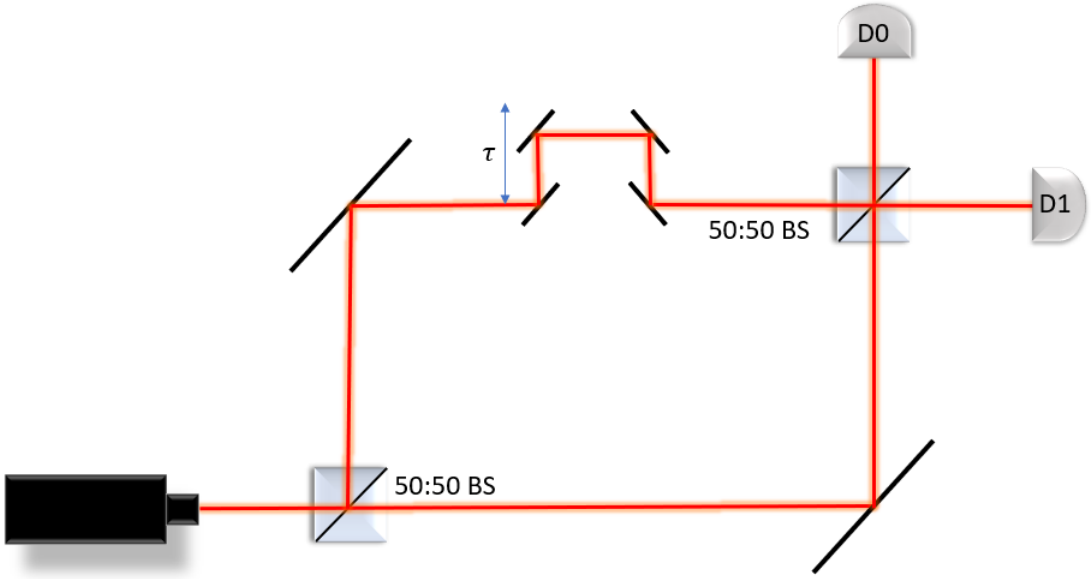

Figure 2.2: A typical Mach-Zehnder interferometer with a variable delay line of time τ.

The quality of the interference fringes of an interferometer can be measured using the *interference visibility*, given by:

$$V = \frac{I_{\max}(\tau) - I_{\min}(\tau)}{I_{\max}(\tau) + I_{\min}(\tau)}, \tag{2.6}$$

where $I_{\max}(\tau)$ and $I_{\min}(\tau)$ are the maximum and minimum signals obtained at a detector within the neighborhood of a given τ, respectively. The interference visibility ranges from 0 (no interference) to 1 (perfect interference). Suppose that the intensity of the signal at detector D0 for time delay τ in Fig. 2.2 is monitored. Then one can simply perturb τ slightly until maximum and minimum signal is obtained in order to calculate the interference visibility. When $\tau > \tau_c$, there will be equal intensities measured at $D0$ and $D1$ (no interference) and the visibility would be 0. In any practical experiments involving interferometers, τ is set such that maximum interference visibility occurs. This time delay corresponds to optimal coherence between the interfering beams.

Chapter 3

Generalised Probabilistic Theories

This chapter provides an introduction to generalised probabilistic theories (GPTs) and the mathematical framework that will be applied to the data obtained from the experiment described in chapter 4. We start with an introduction of the GPT framework and its relevance within the field of quantum foundations. In section 3.2, we discuss the mathematical details of the GPT formalism, including the basic framework, dual spaces, and the no-restriction hypothesis. We then represent qutrit quantum mechanics in terms of the GPT framework as an illustrative example. In section 3.3, we discuss the implications of conducting an experiment in the GPT framework. Specifically, we discuss the GPT inference problem. The content in this chapter will be presented in a theory neutral manner and will construct the GPT framework for an n-level system. In chapter 4, we apply these methods to analyze an experiment conducted on a three-level ($n = 3$) system.

3.1 Introduction

In spite of the fact that quantum theory has so far been quite successful in describing nature, it may one day be replaced by a novel post-quantum theory. In recent years, there has been progress towards identifying potential candidates for such a theory [45, 33, 46, 15]. Some of these alternative theories can be explained within the quantum formalism itself, such as models that exhibit intrinsic decoherence [19, 43, 52, 37]. However, most alternate theories require an outright rejection of some or all of the quantum framework. Some pertinent examples include Almost Quantum Theory [39, 47], theories involving higher order interference [51, 50, 41, 27], and theories involving quaternions [44, 1].

Although there exists a vast amount of theoretical proposals, there is a lack of experimental data that can test the validity of these alternate theories. Experimentally testing alternate theories of nature is difficult as one cannot assume the validity of any one thoery when conducting the experiment, or when analyzing the resulting data. For this reason, a more general framework must be adopted to analyze the experimental data obtained. One of the best existing tools for achieving a theory-neutral lens when examining experimental data is the framework of *Generalized Probabilistic Theories* (GPTs).

The GPT formalism provides a description of a physical system from an operational approach [26, 23, 13, 5, 22, 48, 55, 18, 49, 33] and is used frequently in the field of quantum foundations. It is operational because it describes the theory based solely on its predictions for the probabilities of each outcome of a measurement in an experiment. Using this framework requires the admission of two weak assumptions, both of which are adopted in typical quantum mechanical experiments. The first is that the choice of preparation settings is made independently from the choice of measurements. The second assumption is that repeated runs of identical experiments yields an independent and identically distributed source of data. Under the validity of these two assumptions, the GPT framework is entirely general when used to model experimental data, allowing one to avoid any biases towards quantum theory.

Recent work placed bounds on possible deviations from quantum theory within a landscape of alternatives for a two-level system using the GPT framework [34]. In that work, the two-level system was entirely described by the polarization degree of freedom of a single photon. Specifically, single photons were prepared in a number of polarization states and many distinct measurements were performed on each preparation. The authors developed a new type of tomography which requires no prior knowledge of either the preparation or measurement procedures. This proved advatageous in that it required minimal assumptions to be made in advance of running the experiment. Instead, this tomographic scheme characterizes the preparations and measurements simultaneously based on their interaction. They coined this method *self-consistent tomography*.

In this chapter, we outline the self-consistent tomography scheme that we use to conduct the experiment on our photonic three-level system discussed in chapter 4.

3.2 Details of the GPT Framework

3.2.1 Fundamentals

In any physical theory, there are a multitude of ways in which a system can be prepared, transformed, and measured. The various combinations of these three fundamental components induces a set of possible experimental configurations. For the purposes of this thesis, we will focus on configurations of the prepare/measure variety, meaning that the transformation component is split and absorbed into the preparation and measurement procedures. In addition, we assume that the measurements conducted are terminal so that the system is destroyed after measurement.

A GPT aims to describe the operational phenomenon of an experiment. In situations such as the prepare/measure experiment that will be outlined in chapter 4, a GPT aims to describe the relative probabilities of each possible outcome, given a combination of preparation and measurement settings. Suppose that an experiment has N possible outcomes associated with the measurement procedures. Then a GPT can characterize the experiment using only $N-1$ outcomes as the remaining outcome is determined by normalization. Furthermore, we can convert any N-outcome measurement into a binary-outcome measurement by binning $N-1$ outcomes together as one. Following this realization, we conclude that it is sufficient to record only the event occurring at one of the outcomes of an experiment in order to gain a complete depiction of the underlying GPT. Here, we make the conventional choice to define outcome 0 as the sole outcome of interest when constructing a GPT.

Let m be the number of preparation settings P in an experiment. Similarly, let n be the number of measurement settings M for an experiment. Then the GPT need only specify the probabilities corresponding to outcome 0, denoted $p(0|P_i, M_j)$ for each preparation/measurement combination (P_i, M_j), were $i \in \{1, 2, \ldots, m\}$ and $j \in \{1, 2, \ldots, n\}$. The full set of probabilities obtained from every possible combination of preparations and measurements can be organized into an $m \times n$ matrix D, where each row represents all possible measurements acting on a single preparation, and each column represents all possible preparations being acted on by a single measurement. That is, D takes on the form

$$D = \begin{pmatrix} p(0|P_1, M_1) & p(0|P_1, M_2) & \ldots & p(0|P_1, M_n) \\ p(0|P_2, M_1) & p(0|P_2, M_2) & \ldots & p(0|P_2, M_n) \\ \vdots & \vdots & \vdots & \vdots \\ p(0|P_m, M_1) & p(0|P_m, M_2) & \ldots & p(0|P_m, M_n) \end{pmatrix}. \tag{3.1}$$

D is called the probability matrix associated with the GPT, and it encapsulates all information governing the underlying theory. Let k be the rank of the matrix D. Then it is possible to decompose D into two rectangular matrices

$$D = SE, \tag{3.2}$$

where S is an $m \times k$ matrix and E is a $k \times n$ matrix. The rows of S are called *GPT state vectors*, denoted $\boldsymbol{s_i}$, and the columns of E are called *GPT effect vectors*, denoted $\boldsymbol{e_j}$. With this notation, D can be rewritten as

$$D = \begin{pmatrix} \boldsymbol{s}_1^T \\ \boldsymbol{s}_2^T \\ \vdots \\ \boldsymbol{s}_m^T \end{pmatrix} \begin{pmatrix} \boldsymbol{e}_1 & \boldsymbol{e}_2 & \dots & \boldsymbol{e}_n \end{pmatrix}, \tag{3.3}$$

so that

$$p(0|P_i, M_j) = \boldsymbol{s}_i \cdot \boldsymbol{e}_j. \tag{3.4}$$

In this way, every experimental preparation setting has an associated k-dimensional GPT state vector and each experimental measurement setting has an associated k-dimensional GPT effect vector. A GPT is fully specified by the sets of all physically allowed state and effect vectors , denoted $\boldsymbol{\mathcal{S}}$ (GPT state space) and $\boldsymbol{\mathcal{E}}$ (GPT effect space), respectively.

Every GPT effect vector lives in a k-dimensional vector space, implying that for m effect vectors, only k of them are linearly independent. We call a set of k linearly independent state/effect vectors a tomographically complete set. It follows that in order to fully characterize a single GPT state vector, one must perform at least k measurement configurations corresponding to a tomographically complete set of GPT effect vectors. In a similar manner, in order to characterize a particular GPT effect vector, one needs to apply the measurement configuration to at least k linearly independent state vectors that comprise a tomographically complete set.

It is important to make the distinction that the decomposition $D = SE$ is not unique. In fact, any invertible $k \times k$ matrix A can be inserted into the decomposition such that $D = SAA^{-1}E$. Consequently, the GPT state and effect vectors depend directly on the specific decomposition chosen. For any choice of decomposition, the matrix SA corresponds to a set of valid GPT state vectors, and the matrix $A^{-1}E$ corresponds to a set of valid GPT effect vectors. Furthermore, any two decompositions are related to one another by a linear transformation.

If one makes the weak assumption that any convex mixture of valid preparations is also a valid preparation, then the GPT framework provides a complete description of the underlying theory. That is to say, the GPT is a description with the fewest number of parameters possible. Under this assumption, we can deduce that for all the GPT state vectors admitted by the theory, all vectors in their convex hull are also admitted in the theory. Therefore, the GPT state space $\boldsymbol{\mathcal{S}}$ is a convex set. We make the analogous assumption for the GPT effect space $\boldsymbol{\mathcal{E}}$. Namely, that any convex mixture of measurements is also another valid measurement. In this way, the GPT effect space consists of the convex hull of all GPT effect vectors. In addition, for every valid GPT effect vector $\boldsymbol{e}$, its complement effect $\bar{\boldsymbol{e}} = \boldsymbol{u} - \boldsymbol{e}$ is also a valid GPT effect vector. The vector $\boldsymbol{u}$ is called the unit effect and admits the form $\boldsymbol{u} = (1, 0, 0, \ldots, 0)^T$ in the GPT. The inner product of any GPT state vector with the unit effect yields a probability of 1. The convention is chosen so that the first measurement effect $\boldsymbol{e}_1$ in the GPT is the unit effect, which forces the first columns of the matrices S and D to be filled with ones. For this reason, each k-dimensional GPT state vector can be fully described by $k - 1$ parameters.

We end this section by noting that GPTs in which the assumptions made above do not hold (GPTs which are not closed under convex mixtures) are unlikely to describe the true GPT underlying nature. This is because the most current evidence supports the fact that the GPT describing nature supports classical probability theory as a subtheory, and must therefore support the implementation of arbitrary convex mixtures. In chapter 4, we will examine the shapes of the convex GPT state and effect spaces that are constructed directly from experimental data and use these results to infer bounds on the true GPT underlying nature.

3.2.2 Dual spaces and the No-Restriction Hypothesis

A state vector is said to be *logically* possible if its inner product yields a valid probability with any effect allowed by the GPT. That is, the set of logically possible states, denoted $\boldsymbol{\mathcal{S}}^{\text{logical}}$, consists of all $\boldsymbol{s}$ such that $\forall \boldsymbol{e} \in \boldsymbol{\mathcal{E}} \ : \ 0 \leq \boldsymbol{s} \cdot \boldsymbol{e} \leq 1$ and $\boldsymbol{s} \cdot \boldsymbol{u} = 1$. The set of logically possible states are therefore defined as the intersection of the geometric dual of $\boldsymbol{\mathcal{E}}$ and the hyperplane $\boldsymbol{s} \cdot \boldsymbol{u} = 1$. For simplicity, we will from now on refer to $\boldsymbol{\mathcal{S}}^{\text{logical}}$ as the dual of $\boldsymbol{\mathcal{E}}$, $\boldsymbol{\mathcal{S}}^{\text{logical}} \equiv \text{dual}(\boldsymbol{\mathcal{E}})$. In an analogous way, we define the logical effects as the set of all $\boldsymbol{e}$ such that $\forall \boldsymbol{s} \in \boldsymbol{\mathcal{S}} \ : \ 0 \leq \boldsymbol{s} \cdot \boldsymbol{e} \leq 1$ and denote it as $\boldsymbol{\mathcal{E}}^{\text{logical}}$. In reality, $\boldsymbol{\mathcal{E}}^{\text{logical}}$ is the geometric dual of the set of sub-normalized states (let $w \in [0, 1]$, then the set of sub-normalized states is simply $w\boldsymbol{s}$ for all $s \in \boldsymbol{\mathcal{S}}$), but for simplicity, we define $\boldsymbol{\mathcal{E}}^{\text{logical}} \equiv \text{dual}(\boldsymbol{\mathcal{S}})$.

Theories in which $\boldsymbol{\mathcal{S}}^{\text{logical}} = \boldsymbol{\mathcal{S}}$ and $\boldsymbol{\mathcal{E}}^{\text{logical}} = \boldsymbol{\mathcal{E}}$ are said to satisfy the *no-restriction hypothesis* [14]. For GPTs satisfying the no-restriction hypothesis, every logically allowed

GPT state vector corresponds to a physically realizable GPT effect vector and vice versa. Theories in which the no-restriction hypothesis is violated contain some state (effect) vectors that are non-physical, but still assign valid probabilities when paired with any physically allowed effect (state) vector.

3.2.3 Example: Qutrit Quantum Mechanics

One of the most familiar examples of a GPT is quantum mechanics itself. Although a GPT can be constructed for any dimensional system we desire, we will focus here on qutrit quantum mechanics as it is most relevant to this thesis.

The GPT state space $\mathcal{S}$ in this case is the set of all 3×3 positive semi-definite trace one operators ρ with Bloch decomposition given by equation (1.18). The GPT state vectors comprising the rows of S are the Bloch coefficients of every ρ. That is, $\boldsymbol{s} = (1, s_1, s_2, \ldots, s_8)$. The GPT effect space $\mathcal{E}$ is given by all positive semi-definite operators Q less than identity $0 \leq Q \leq I$ and with Bloch decomposition matching equation (1.32). The GPT effect vectors comprising the columns of E are the Bloch coefficients of every Q. That is, $\boldsymbol{e} = (e_0, e_1, \ldots, e_8)$.

Defining the GPT state and effect vectors in this way implies that

$$D_{ij} = p(0|P_i, M_j) = \boldsymbol{s}_i \cdot \boldsymbol{e}_j = \mathrm{Tr}[\rho_i Q_j], \tag{3.5}$$

and that the rank of the matrix D in this case is $k = 9$ since the matrices S and E contain state and effect vectors that are 9-dimensional. The geometry of the GPT state and effect spaces for qutrit quantum mechanics is represented by the projections shown in Fig. 1.2. The definition of D in equation (3.1) is shown with a finite number of preparations, m, and a finite number of measurements, n. In the case of quantum mechanics, the set of all GPT states and effects is infinite, meaning $m = n = \infty$. However, any practical experiment can only ever characterize a finite number of states/effects. As long as the number of states and effects characterized by the experiment is dense enough, the resulting GPT state and effect spaces should be a reasonable approximation to the true GPT that would be achieved from infinite run-statistics.

Quantum mechanics satisfies the no-restriction hypothesis, meaning that every logically allowed quantum state corresponds to a physical quantum measurement effect and vice versa. From a geometric perspective, this dictates that the full nine-dimensional GPT state and effect spaces are exactly the geometric duals of each other. Although Figure 1.2 shows matching pairs of projections in the quantum state and effect spaces, these

projections are not geometric duals of one another. The duality condition only holds on the full-dimensional spaces. If the true GPT describing nature is indeed quantum theory, we would expect that any GPT state and effect spaces realized by an experiment will be geometrically similar to the state and measurement spaces predicted by quantum mechanics (have similar looking 3-dimensional projections as those shown in Figure 1.2).

3.3 The GPT Inference Problem

For many theories (including quantum mechanics), the GPT state and effect spaces $\boldsymbol{\mathcal{S}}$ and $\boldsymbol{\mathcal{E}}$ consist of a continuum of states and effect vectors. Thus, from an experimental perspective, these spaces are purely theoretical, representing what could be measured with infinite sampling and no noise. In reality, any reasonable experiment can only sample some finite number of preparation and measurement settings before running into time and computational restraints. Furthermore, an experimentalist can never remove all of the noise associated with a physical system. Both of these reasons imply that the convex hull of experimentally realized GPT state and effect spaces, which we denote $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{E}}^{\text{realized}}$, respectively, will be contracted with respect to their theoretical counterparts. That is, $\boldsymbol{\mathcal{S}}^{\text{realized}} \subset \boldsymbol{\mathcal{S}}$ and $\boldsymbol{\mathcal{E}}^{\text{realized}} \subset \boldsymbol{\mathcal{E}}$. In section 3.2.1, we made the weak assumptions that the GPT state and effect spaces are convex. Thus, for a finite set of experimental preparation and measurement settings, $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{E}}^{\text{realized}}$ will be defined as the convex hulls of the realized GPT state and effect vectors. As long as the experiment characterizes a sufficiently dense set of preparation and measurement settings, the resulting convex polytopes should resemble their theoretical abstractions.

The space $\boldsymbol{\mathcal{S}}^{\text{realized}}$ defines the set of GPT effect vectors that are logically consistent with the experimentally realized preparations. We denote this set $\boldsymbol{\mathcal{E}}^{\text{consistent}}$ so that $\boldsymbol{\mathcal{E}}^{\text{consistent}} \equiv \text{dual}(\boldsymbol{\mathcal{S}}^{\text{realized}})$. Similarly, $\boldsymbol{\mathcal{E}}^{\text{realized}}$ defines the set of GPT state vectors that are logically consistent with the experimentally realized measurements. We denote this set as $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ so that $\boldsymbol{\mathcal{S}}^{\text{consistent}} \equiv \text{dual}(\boldsymbol{\mathcal{E}}^{\text{realized}})$. Since the experimentally realized GPT spaces are necessarily subsets of the theoretical GPT spaces, the consistent GPT spaces are necessarily supersets of the logical GPT spaces. That is, $\boldsymbol{\mathcal{S}}^{\text{logical}} \subset \boldsymbol{\mathcal{S}}^{\text{consistent}}$ and $\boldsymbol{\mathcal{E}}^{\text{logical}} \subset \boldsymbol{\mathcal{E}}^{\text{consistent}}$.

If one obtains the realized GPT state and effect spaces $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{E}}^{\text{realized}}$ from an experiment, they can use it to infer knowledge about the true GPT spaces $\boldsymbol{\mathcal{S}}$ and $\boldsymbol{\mathcal{E}}$ governing nature. The key insight is as follows: The true GPT state and effect spaces can be no smaller than the realized GPT spaces, nor can they be any larger than the logically consistent spaces $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ and $\boldsymbol{\mathcal{E}}^{\text{consistent}}$. Therefore, the true GPT state and effect spaces must lie somewhere in between. The gap between the realized and consistent spaces places

restrictions on the set of possible GPTs underlying nature. As the gap between the realized and consistent spaces increases, so does the set of candidate GPTs that are consistent with the experimental data.

It is important to mention that a large gap between the realized and consistent GPT spaces can be explained in two ways. Either the true GPT underlying nature violates the no-restriction hypothesis to the degree inferred by the gap, or the experiment did not sample a sufficiently dense set of experimental configurations to get an accurate depiction of the true GPT spaces. For this reason, it is vitally important that one avoids undersampling configurations so that the realized GPT spaces closely approximate the true GPT spaces. It is also important that the experimental configurations chosen come as close to uniformly sampling the GPT state and effect spaces as possible. In the experiment outlined in chapter 4, we ensure this by drawing our sets of preparation and measurement settings from a uniform Haar-distribution in accordance with section 1.4.1.

Chapter 4

Self-Consistent Tomography

This chapter outlines the results of the self-consistent tomography scheme applied to a photonic three-level system. Section 4.2 describes the experimental setup and methods. Section 4.3 presents the experimental results. An article has been submitted for publication in *Physical Review A* based on the content contained within this chapter. The contributions to this work are as follows:

- **Kevin Resch** and **Robert Spekkens** proposed experimentally characterizing a three-level system within the framework of generalized probabilistic theories.
- **Tim Hill, Elie Wolfe, Patrick Daley, Aldo Pasos** and **Satchel Armena** performed some preliminary work and relevant simulations relating to the experiment.
- **Michael Grabowecky** and **Christopher Pollack** built the experiment.
- **Michael Grabowecky, Christopher Pollack** and **Andrew Cameron** troubleshooted the experimental setup.
- **Michael Grabowecky** conducted the experiment and performed the data analysis.
- **Michael Grabowecky** is the sole author of the content contained in this chapter.

4.1 Introduction

The constraints describing the geometry of the state and effect spaces of three-level systems are much more complicated in comparison with their two-level counterparts. Compare for

example, the complexity of the derivations of the state and effect spaces for qubit/qutrit quantum mechanics in sections 1.2 and 1.3, respectively. Due to this added complexity, there is a larger potential for an experiment to reveal some surprising discrepancies between nature and the predictions of quantum theory. Specifically, there are two ways in which the experimentally realized GPT representing a three-level system may admit differences from the quantum mechanical predictions. First, the experimental data might find a deviation in the dimension of the GPT vector space relative to quantum theory. In the self-consistent tomography scheme, this would present itself as a deviation in the rank k of the probability matrix D representing the experimental GPT from the $k = 9$ predictions of qutrit quantum theory. Second, if the experimental GPT finds no dimensional deviation, then it may find a deviation in the shape of the state and effect spaces relative to quantum theory. For example, the experimetnal GPT may admit 3-dimensional projections that are vastly different in shape than those presented in Figure 1.2 for qutrit quantum theory.

Since we are not assuming the correctness of quantum theory, the set of preparations and measurements that we implement might fail to be tomographically complete for the three-level system of interest. That is, the conditions for tomographic completeness in our experimentally realized GPT may differ from those of standard quantum theory. In addition, the preparations and measurements required to achieve tomographic completeness might only be implementable in a different experimental set-up from the one we consider in this chapter. Specifically, tomographic completeness may only be achievable in a set-up that involves exotic physics. Because of this, a dimensional discrepancy between the true GPT and quantum theory might exist but be missed by our experiment. However, if a dimensional discrepancy exists, it is possible that it could be detected in a nonexotic experimental set-up (such as ours) because evidence for exotic physics can arise in conventional setups as long as the experiment is conducted at a high enough precision. In this way, the experiment we conduct in this chapter provides an opportunity for uncovering such a discrepancy. Therefore, if no dimensional discrepancy is found in our experiment, one of the following scenarios must be true. Either it does not exist. It exists but can only be detected in the sort of experiment we have implemented at higher precision than we achieved. Or finally, it exists, but can only be detected in a different experimental set-up involving exotic physics.

Assuming that there is no dimensional discrepancy, we can infer the geometry of the realized state and effect spaces from our experimental data using the self-consistent tomography scheme, and from these we can compute the geometry of the logically consistent state and effect spaces. Together, these yield inner and outer bounds for the state and effect spaces of the true GPT governing the three-level system. By doing this, we can place quantitative bounds on the magnitude of any discrepancies between the *shapes* of

the state and effect spaces of the true GPT and those predicted by quantum theory, in particular on the possibility of a violation of the no-restriction hypothesis. For a more in-depth discussion of what we can infer about possible discrepancies in the dimension and shapes of the GPT state and effect spaces, we refer the reader to the introduction of Ref. [34].

In this chapter, we conduct an experiment on a photonic three-level system within the GPT framework. Our experiment is conducted using heralded single photons prepared using spontaneous parametric down conversion (SPDC). Three-level states are encoded in three modes of the single photons which are distinguished by their polarization and spatial degrees of freedom. We perform the self-consistent tomography scheme outlined in chapter 3 on a large set of preparations and a large set of measurements to obtain a GPT characterization of both. The analysis of our experiment uses a model selection technique in which data is collected for two independent runs of the experiment. The first data set obtained is called the *training set* and the second data set obtained is called the *test set*. For each of a set candidate dimensions, we find the corresponding GPT characterization that best fits the data in the training set. We then rate each of the characterizations based on how well they predict the data in the test set. This differs from the model selection technique used in Ref. [34] (which relied on the Akaike Information Criterion [2], and in which the authors applied the self-consistent tomography analysis on a two-level photonic system) in that it avoids certain assumptions regarding the error model chosen for the experimental data.

4.2 Description of the Experiment

4.2.1 Heralded Single Photon Source

For our experiment, we require a source of single photons. We utilize a single photon source constructed in a linear configuration. This section describes its functionality. For a good introduction to non-linear optics, see Ref. [9]. For more information regarding the mechanics governing single photon sources that utilize SPDC, see Ref. [25].

A diagram of the heralded single photon source is shown in Figure 4.1. A photo of the source is shown in Figure 4.2. A continuous wave laser (Toptica TopMode) at 404 nm is focused using a 150 mm lens onto a periodically poled potassium titanyl phosphate (PPKTP, Raicol Crystals Ltd. 404 nm) crystal which creates pairs of degenerate 808 nm orthogonally polarized single photons via type-II SPDC. The laser used to pump the crystal

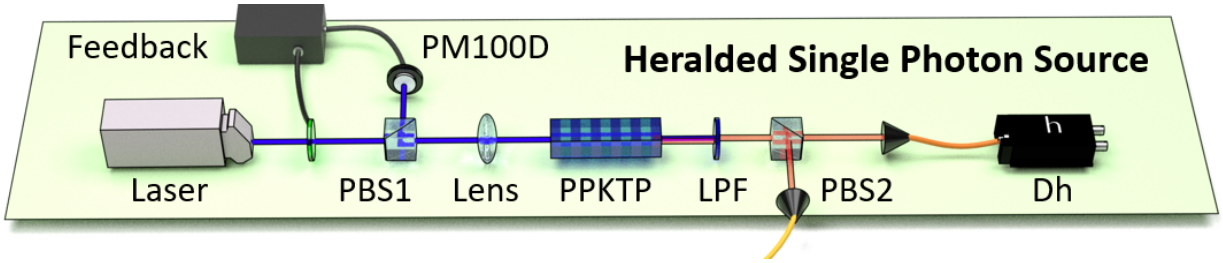

Figure 4.1: Single photons at 808 nm are generated via type-II SPDC using a continuous wave laser diode at 404 nm focused onto a PPKTP crystal using a lens. Polarization drifting effects inherent to the laser are actively compensated using a power meter (PM100D) connected via a feedback loop to a motorized HWP. Excess 404 nm light is filtered out with a long pass filter (LPF). Photons detected at APD *Dh* are used to herald the arrival of the signal photons passed to the remainder of the experimental setup.

exhibits an inherent polarization drifting effect, causing the output polarization of the laser to vary over time. We require that the pump photons be as close to the computational basis state $|H\rangle$ as possible to ensure that the efficiency of the SPDC is optimal. Thus, it is necessary to compensate for the polarization drifting effect of the source to ensure constant crystal pump polarization at all times. We compensate the polarization drifting by placing a power meter (PM100D) in the reflected port of a PBS1 (positioned directly after the laser). This power meter is connected via a feedback loop to a motorized HWP (Casix Zero Order Wave Plate 404 nm) which actively compensates for the polarization drifting. All other HWPs/QWPs used in this experiment are Casix Airspace Zero Order Wave Plates specified to 810 nm.

A long pass filter (LPF) is used to remove any excess 404 nm light that is not extinguished when passing through the crystal and a second PBS splits the output photons based on their polarization. The herald photon is immediately sent to an avalanche photo diode (APD, SPCM-AQ4C from Excelitas Technologies) and is used to herald the arrival of the signal photons travelling through the remainder of the experimental setup within a coincidence window of 3 ns.

4.2.2 Displaced Sagnac Interferometer

We construct the three-level photonic system by exploiting the polarization and spatial degrees of freedom of single photons. We access these degrees of freedom using a Mach-Zehnder interferometer in a displaced Sagnac configuration. This configuration differs from

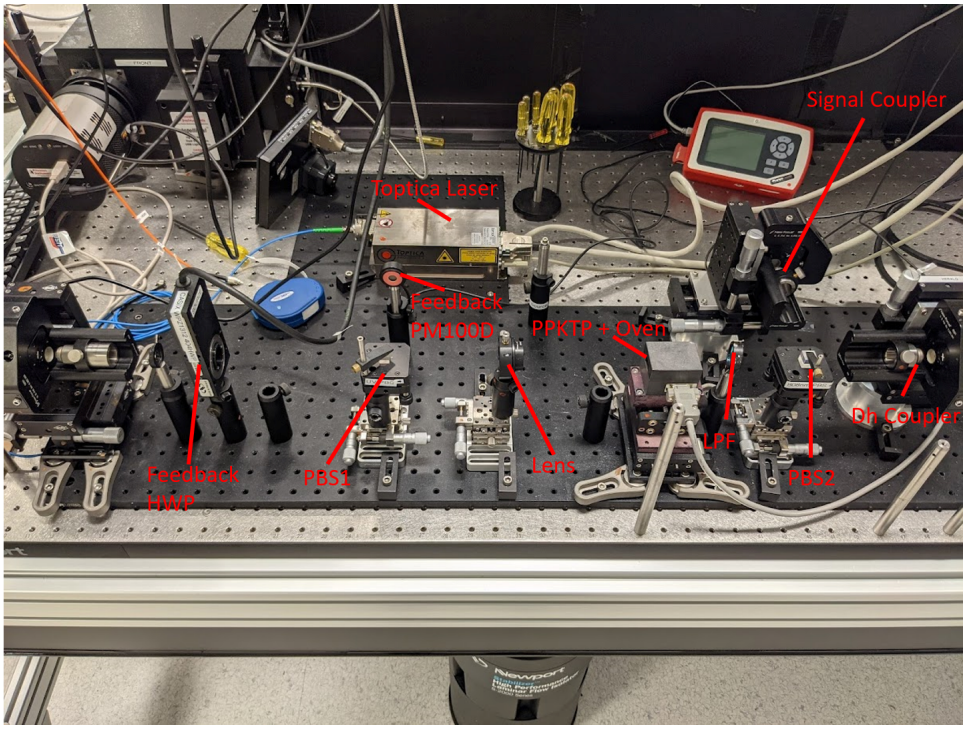

Figure 4.2: Labeled photo of the heralded single photon source used for our experiment.

the standard Mach-Zehnder configuration discussed and shown in Figure 2.2, in that only one beam splitter is utilized. Such a configuration is well known to exhibit a high degree of inherent stability compared with other types of Mach-Zehnder interferometers (see [38] and references therein). This inherent stability is due to the fact that both interfering fields transverse the same optical paths, but in opposing directions. The displaced nature of the interferometer is utilized to address each optical path individually to control the independent modes of the three-level system.

The full experimental setup is shown in Figure 4.3. The setup consists of two displaced Sagnac interferometers in sequence. A labelled photo of the interferometers is shown in Figure 4.4. The first interferometer is used to prepare the three-level system, and the second interferometer is used to measure the three-level system.

Single photons from the source are coupled into a single-mode fiber with polarization control (PC). A calcite beam displacer (CBD) is used with the PC to ensure that vertically polarized light is incident to the interferometers at all times. Different states of the three-level system are prepared using motorized quarter wave plate (QWP) and HWP pairs

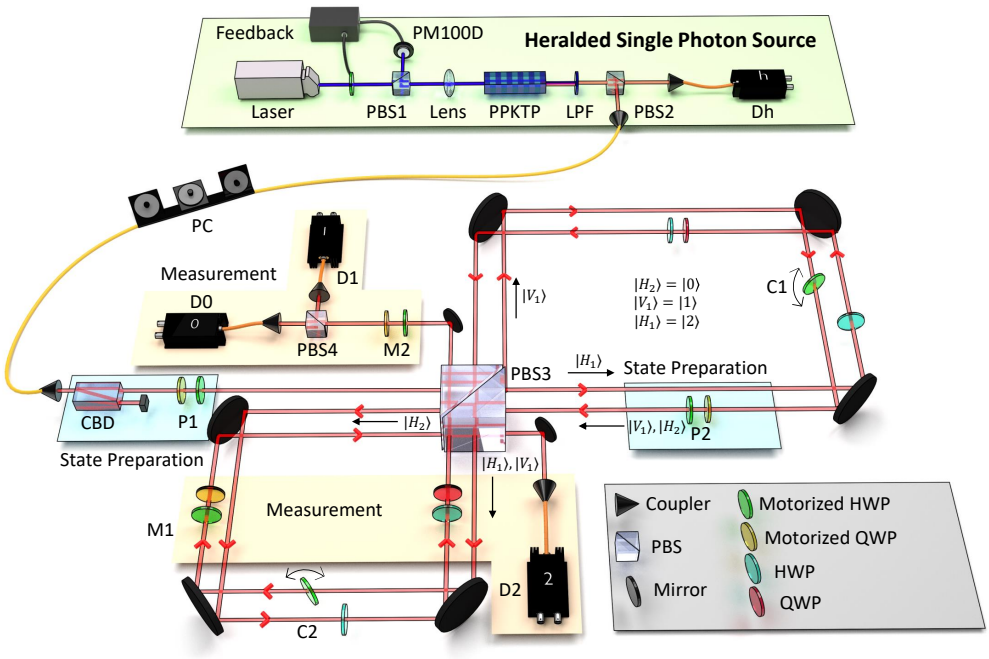

Figure 4.3: Single photons at 808 nm are generated via type-II SPDC using a continuous wave diode laser operating at 404 nm focused onto a PPKTP crystal with a lens. Polarization drifting effects of the source are actively compensated using a power meter (PM100D) connected via a feedback loop to a motorized HWP. Excess 404 nm light is filtered out with a long pass filter (LPF). Photons detected at APD Dh herald the arrival of signal photons arriving at APDs $D0$ (outcome 0), $D1$ (outcome 1), and $D2$ (outcome 2) through the displaced Sagnac interferometer with a coincidence window of 3 ns. Three-level states are prepared in the modes $|H_1\rangle$, $|V_1\rangle$, and $|H_2\rangle$ using QWP/HWP pairs P1, P2 (highlighted in blue) and are measured using HWP/QWP pairs M1, M2 (highlighted in orange). The interference visibility is actively compensated using tilted HWPs $C1$ and $C2$ which are placed on motorized mounts. The arrows shown within the interferometer arms are displayed for clarity of photon direction.

P1 and P2 which are highlighted in blue in Figure 4.3. Preparation stage P1 is used to prepare a two-level state with polarization modes $|H_1\rangle$ and $|V_1\rangle$. These modes are separated on a 1 in polarising beam splitter (PBS3). The horizontally polarized mode $|H_1\rangle$ travels counterclockwise around the first interferometer loop and returns to PBS3 unchanged. The vertically polarized mode $|V_1\rangle$ travels clockwise around the first interferometer loop and is

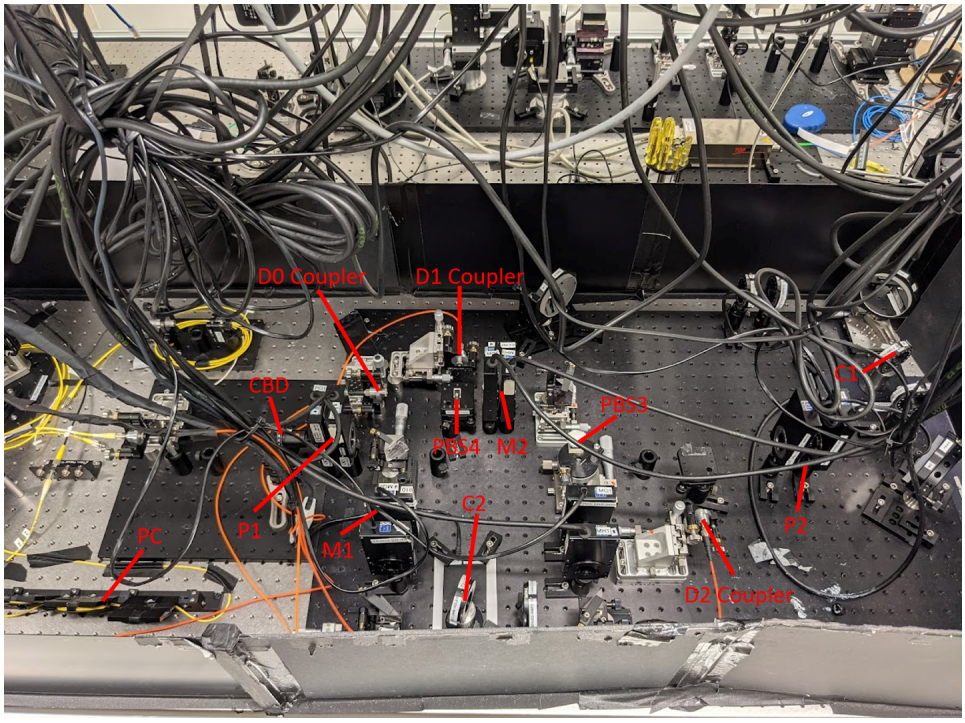

Figure 4.4: Labeled photo of the displaced Sagnac interferometers.

addressed by preparation stage P2 where the third mode of the three-level state, denoted $|H_2\rangle$, is prepared. Each three-level state is fully constructed once all three modes recombine on PBS3. The prepared three-level states are then measured using HWP/QWP pairs M1, M2, and are detected at one of three APDs D0, D1, or D2. All measurement components are highlighted in orange in Figure 4.3. The non-motorized waveplates in Fig. 4.3 (shown in blue and red) are used to balance the optical path lengths of the interferometer arms and always have optical axis oriented along the horizontal.

The interferometers are displaced to approximately 5mm. This displacement was chosen to avoid clipping the beam on the various optical components. Each of the wave plates within the interferometer arms have 3.5mm holes so that one of the two beams can pass freely through without interaction. This ensures that each of the beams can be addressed independently from one another. The arrows along the beam paths in Figure 4.3 are shown for clarification of photon direction.

4.2.3 Active Relative Phase Compensation

Due to air fluctuations, vibrations within the optical components, and temperature changes, the optical path length difference between the arms of the interferometers changes over time. These path length changes result in fluctuations in the relative phases between the beams in the interferometer arms over time [38], which degrades their coherence, and ultimately decreases the quality of the interference visibility (as per the discussion in section 2.2). Degradation of the interference visibility directly impacts the quality of the experimental data. Fortunately, there are a variety of methods that one can use to compensate for phase drifting effects. In our experiment, we compensate for relative phase drifting using a tilted HWP in one arm of each of the two interferometers (C1 and C2 in Figure 4.3). These tilted wave plates are pivoted about the axis perpendicular to the beam using motorized mounts to precisely control the phase delay caused by the wave plate glass.

During the experiment, the relative phase of each of the interferometers is checked and corrected once every 20 minutes. This timescale is chosen as it is observed that the relative interferometer phase typically begins to drift after this length of time. When the phase is checked, each of the interferometers are configured (one at a time) so that maximum interference would occur at PBS3 and such that minimum signal would be detected at APD D1. A program then monitors the counts obtained at D1 as it actively pivots the wave plate C1 (or C2 depending on the interferometer being compensated) until a threshold count rate is obtained. At this point, the last experimental configuration from before the active compensation program ran is recovered and the experiment continues for another 20 minutes.

To illustrate the operation of the active compensation scheme, we ran the program continuously for 20 h with classical light incident to the interferometers. The single photon detectors were disconnected for the duration of the test. A power meter was used to measure the beam intensity at D1. The total power observed was 420 μW and the target power threshold at D1 was set to 16 μW, corresponding to a threshold interference visibility of 0.93. We ran this test on the second interferometer and actively monitored the tilt angle of the wave plate C2. Figure 4.5 compares the power measured at $D1$ to the relative tilt angle of C2 (from starting position) over the full 20 h. It is clear that the active compensation was successful in keeping the interference visibility above threshold for the entirety of the testing period with only small changes in wave plate tilt angle. Specifically, the relative tilt angle never changed by more than $\pm 0.14^\circ$ over the full 20 h period.

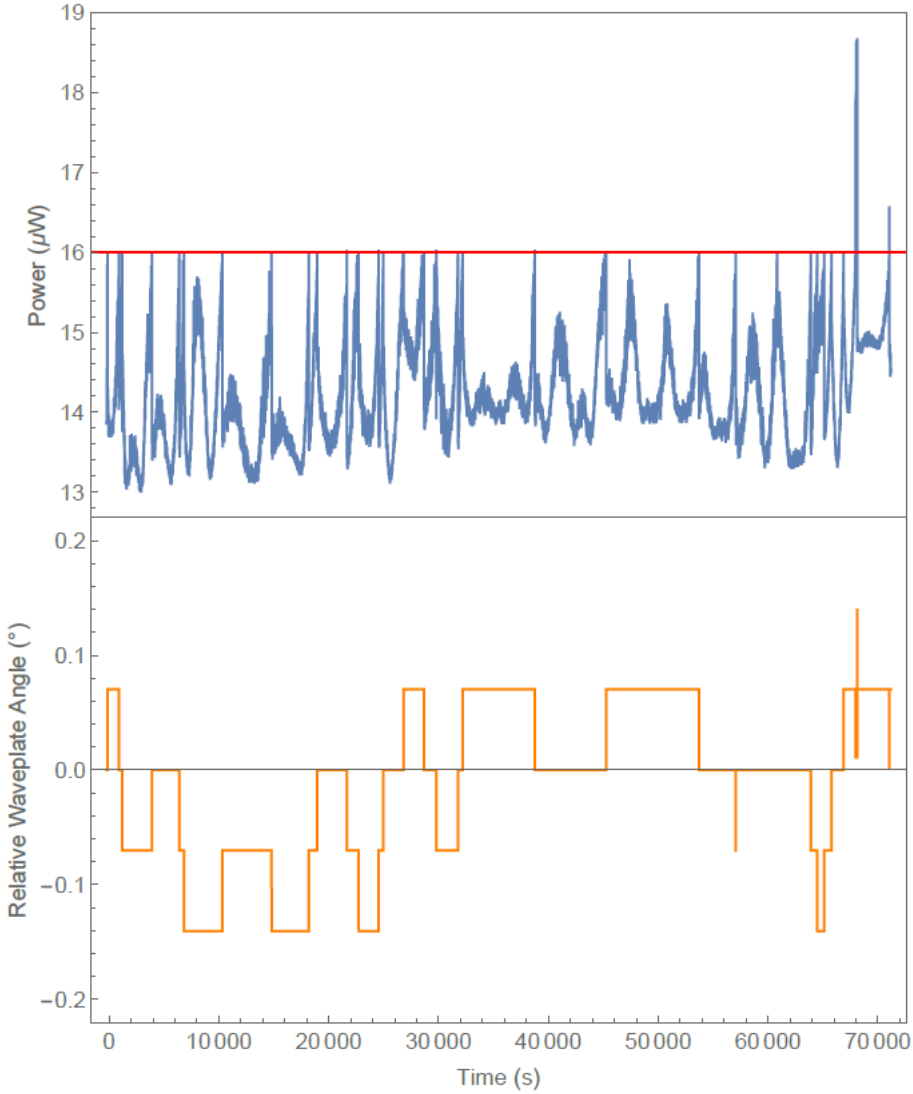

Figure 4.5: Top: Power detected at D1 while the active compensation program was running continuously for 20 h in the second interferometer. The power threshold was set to 16 μW (shown in red). Bottom: Relative tilt angle of C2 while the active compensation program was running. The compensation wave plate never pivoted more than 0.14° from its starting position.

4.2.4 Choosing Preparation and Measurement Settings

The purpose of our experiment is to characterize the GPT state and effect spaces underlying our data as uniformly as possible. For this reason, we choose our experimental preparation settings according to the $SU(3)$ invariant Haar measure and generated a number of random qutrit quantum states using the algorithm outlined in section 1.4.1. By assuming the correctness of quantum theory, we then compute the HWP and QWP angles needed to generate each of these random states given our experimental setup. We accomplish this using a method similar to that described in section 2.3. These angles are then passed to the QWP/HWP pairs P1 and P2 shown in figure 4.3 to prepare each of the Haar-distributed states in our experiment.

To choose our experimental measurement settings, we make use of the fact that for every pure state $\rho_j = |\psi_j\rangle\langle\psi_j|$, there exists a projective effect $Q_j = \rho_j = |\psi_j\rangle\langle\psi_j|$ such that $\mathrm{Tr}[\rho_j Q_j] = \mathrm{Tr}[|\psi_j\rangle\langle\psi_j|\psi_j\rangle\langle\psi_j|] = 1$. Recall from section 3.2 that any GPT need only describe the statistics of a single outcome in a binary outcome measurement. Thus, our measurements are chosen such that $\{Q_j, \bar{Q}_j\} = \{|\psi_j\rangle\langle\psi_j|, I - |\psi_j\rangle\langle\psi_j|\}$, where each $|\psi_j\rangle\langle\psi_j|$ corresponds to one of the Haar-distributed preparation states. Here, Q_j is the effect corresponding to outcome 0 and $\bar{Q}_j$ is the effect corresponding to the combined outcomes 1 and 2. The measurement settings are the wave plate angles that implement the projective measurement $\{Q_j, \bar{Q}_j\}$ when passed to the HWP/QWP pairs M1 and M2 shown in figure 4.3. Although we convert our three-outcome measurement into a binary-outcome measurement in post-processing, during the experiment, we measure photon counts at all three detectors independently. For this reason, there are infinitely many three-outcome POVMs that implement $\bar{Q}_j$ at the combined detectors 1 and 2. However, ensuring that the effect Q_j is implemented at detector D0 yields sufficient information to conduct our GPT analysis scheme. We therefore find any set of wave plate angles that accomplishes Q_j at detector D0, and refrain from imposing any constraints on the effects implemented at detectors D1 and D2. Our choices of measurement settings produce many instances in which $p(0|P, M) = 1$. However in practice, limitations of the experimental setup always bound the probabilities measured away from 1, so that the uncertainties on our counts never produce instances of super-normalized outcomes. It is worth emphasizing that although we assume the correctness of quantum theory when choosing our experimental preparation and measurement settings, we do not assume its correctness in any of our analysis techniques.

For our first experiment, we sample $m = 100$ Haar-distributed preparation settings and $n = 100$ corresponding measurement settings for a total of $mn = 10^4$ experimental configurations. We detect coincidences at an approximate rate of 2000 counts/s and integrate the photon counts of each measurement configuration for 2 s. It takes additional time for

the motorized wave plates to configure between each setting. We mitigate this additional time by ordering the preparation and measurement settings in such a way that the required wave plate rotations between each subsequent setting is minimized. Finally, the extra time needed to actively compensate relative interferometer phase (every 20 minutes) brings the total experimental run-time up to approximately 16 h.

4.3 Results

In this section we present the results of the self-consistent GPT tomography scheme for our experimental photonic three-level system.

4.3.1 Inferring Best Fit Probabilities from Finite Run Statistics

We begin by showing how to infer the probability matrix D directly from the raw experimental data. We detect heralded photon counts at $D0$, $D1$, and $D2$. For these counts, we assume Poissonian uncertainty for all detectors. Since a GPT can be fully characterized from the probabilities associated with one outcome of a binary-outcome measurement, we combine the counts at detectors $D1$ and $D2$ together to form a single outcome. Noisy approximations to the probabilities $p(0|P_i, M_j)$ are obtained by dividing the number of counts detected at $D0$ by the total number of counts recorded at all three detectors. We denote these relative frequencies by $f(0|P_i, M_j)$ and construct a frequency matrix F that is an approximation to the rank-k probability matrix D that describes the photonic system. The experiment was conducted twice (back to back), with each run using identical preparation and measurement settings. We call the first data set obtained the *training set* and the second data set obtained the *test set*. The training set, with frequency matrix F^{train}, is used to find the best-fit rank-k probability matrix D corresponding to our data. Due to experimental noise, the matrix F^{train} will almost always be full rank, regardless of the rank of the probability matrix D underlying the experimental data. Therefore, the problem at hand is to find a matrix D that best fits the matrix of raw frequencies F^{train}, where the rank k is a parameter in the fit. Formally, the task can be written as the weighted low rank approximation problem given by:

$$\begin{aligned} \text{Minimize} \quad & \chi^{2\ \text{train}}_k = \sum_{i=1}^{m}\sum_{j=1}^{n}\left(\frac{F^{\text{train}}_{ij} - D_{ij}}{\Delta F^{\text{train}}_{ij}}\right)^2, \\ \text{Subject To:} \quad & \text{Rank}[D] = k \\ & 0 \leq D_{ij} \leq 1 \end{aligned} \tag{4.1}$$

where $\Delta F_{ij}^{\text{train}}$ is the statistical uncertainty in F_{ij}^{train}.

The column of ones corresponding to the unit effect that we include in F^{train} are added in post and have uncertainty 0. D is defined as the matrix that minimizes $\chi_k^{2\ \text{train}}$, which forces the entries in the same column of D to also be exactly 1. If this was not the case, then $\chi_k^{2\ \text{train}}$ would be undefined for those entries. Solving Eq. (4.1) is an NP-hard problem. However, by using an alternating least squares approach in which D is parametrized as SE, we can convert this problem into a convex quadratic program which can be solved in polynomial time. The algorithm for the alternating least squares approach is as follows: First, we begin with a fixed estimate of the matrix E. We then optimize the matrix S given this estimation. The resulting optimal S matrix is then fixed and the matrix E is varied until an optimum is achieved. This marks the end of one iteration. We repeat this process until the difference in $\chi^{2\ \text{train}}$ values between successive iterations is less than 10^{-4}, or until 1000 iterations is reached.

It has yet to be proven that Eq. (4.1) can be written as a convex quadratic program. To do so, we make use of the vec$(\cdot)$ operator which converts a matrix into a single column vector. The vectorization of a matrix is performed by taking each column and stacking them one on top of another. For example, for an $m \times n$ matrix M, vec(M) is a column vector with the first m elements being the first column of M and the next $m+1$ through $2m$ elements being the second column of M, etc. To represent the uncertainties $(1/\Delta F_{ij}^{\text{train}})$, we define a diagonal matrix of weights W. These weights are ordered in such a way so that we can rewrite $\chi_k^{2\ \text{train}}$ as:

$$\begin{aligned}\chi_k^{2\ \text{train}} &= \text{vec}(F^{\text{train}} - SE)^T W \text{vec}(F^{\text{train}} - SE) \\ &= \text{vec}(SE)^T W \text{vec}(SE) - 2\text{vec}(SE)^T W \text{vec}(F^{\text{train}}) + \text{vec}(F^{\text{train}})^T W \text{vec}(F^{\text{train}}).\end{aligned} \tag{4.2}$$

We then make use of the identity $\text{vec}(SE) = (E^T \otimes I_m)\text{vec}(S)$, where I_m is the $m \times m$ identity matrix. Using this identity, Eq. (4.2) becomes:

$$\begin{aligned}\chi_k^{2\ \text{train}} = \text{vec}(S)^T (E \otimes I_m) W (E^T \otimes I_m)\text{vec}(S) - 2\text{vec}(S)^T (E \otimes I_m) W \text{vec}(F^{\text{train}}) \\ + \text{vec}(F^{\text{train}})^T W \text{vec}(F^{\text{train}}).\end{aligned} \tag{4.3}$$

We can now write the optimization problem over S as

$$\begin{aligned}&\text{Minimize}_S \quad \text{vec}(S)^T (E \otimes I_m) W (E^T \otimes I_m)\text{vec}(S) - 2\text{vec}(S)^T (E \otimes I_m) W \text{vec}(F^{\text{train}}), \\ &\text{Subject To:} \quad 0 \leq (SE)_{ij} \leq 1 \ \forall i, j\end{aligned} \tag{4.4}$$

where the last term of Eq. (4.3) was dropped as it is a constant in the optimization. Because W is a diagonal matrix with strictly positive elements, the matrix $(E \otimes I_m) W (E^T \otimes I_m)$

is positive-definite and Eq. (4.4) is in the precise form of a convex quadratic program [10]. The analogous optimization problem for E can be derived by using the identity $\text{vec}(SE) = (I_n \otimes S)\text{vec}(E)$ in Eq. (4.2) and is written as

$$\begin{aligned} &\text{Minimize}_E \quad \text{vec}(E)^T(I_n \otimes S^T)W(I_n \otimes S)\text{vec}(E) - 2\text{vec}(E)^T(I_n \otimes S^T)W\,\text{vec}(F^{\text{train}}). \\ &\text{Subject To:} \quad 0 \leq (SE)_{ij} \leq 1 \ \forall i, j \end{aligned} \tag{4.5}$$

For a variety of candidate model ranks k, we compute the best fit approximation to the experimental data using the alternating least squares approach. The python 3 code used to solve the alternating least squares optimization is displayed in appendix A.2. We denote the optimal probability matrix obtained for each k as D_k^{realized}. To determine which D_k^{realized} is the best approximation to the true probability matrix D underlying our experiment, we analyze the so called *training error* and *testing error*. The training error for rank k is what we call the $\chi_k^{2\ \text{train}}$ value obtained from substituting D_k^{realized} into equation (4.1) and it is used to determine how much a given model *under-fits* the data. The testing error $\chi_k^{2\ \text{test}}$ for a data set is given by

$$\chi_k^{2\ \text{test}} = \sum_{i=1}^{m}\sum_{j=1}^{n}\left(\frac{F_{ij}^{\text{test}} - D_{k,ij}^{\text{realized}}}{\Delta F_{ij}^{\text{test}}}\right)^2, \tag{4.6}$$

and is used to determine the *predictive power* of a given model, which can be compromised by both under-fitting or *over-fitting* the data. Here, F^{test} is the frequency matrix representing the testing data set and $\Delta F_{ij}^{\text{test}}$ is the statistical uncertainty in F_{ij}^{test}. The rank-k model that minimizes the testing error has the highest *predictive power*, and will be the model accepted as the best fit for our experimental data.

For the 100×100 Haar-distributed experiment, the training and testing error is shown in Figure 4.6. The training error (shown in blue) decreases as the candidate rank k increases from 2 to 9. It continues to decrease as the rank increases from 9 to 12, but at a much slower rate. The testing error (shown in red) also decreases as the candidate rank k increases from 2 to 9. However, unlike the training error, it begins to rise again as the rank increases from 9 to 12. Since the rank-9 model is the one that is found to minimize the testing error, it is the one that has the maximum predictive power. Furthermore, the trends just described are precisely what one expects if the rank-9 model is the transition point between underfitting and overfitting the experimental data, as we explain below.

The set of all probability matrices that can be realized by models of rank $k-1$ is a strict subset of what can be realized by models of rank k because the rank-$(k-1)$ models have strictly fewer parameters than the rank-k models. If it is identified that the rank-$(k-1)$

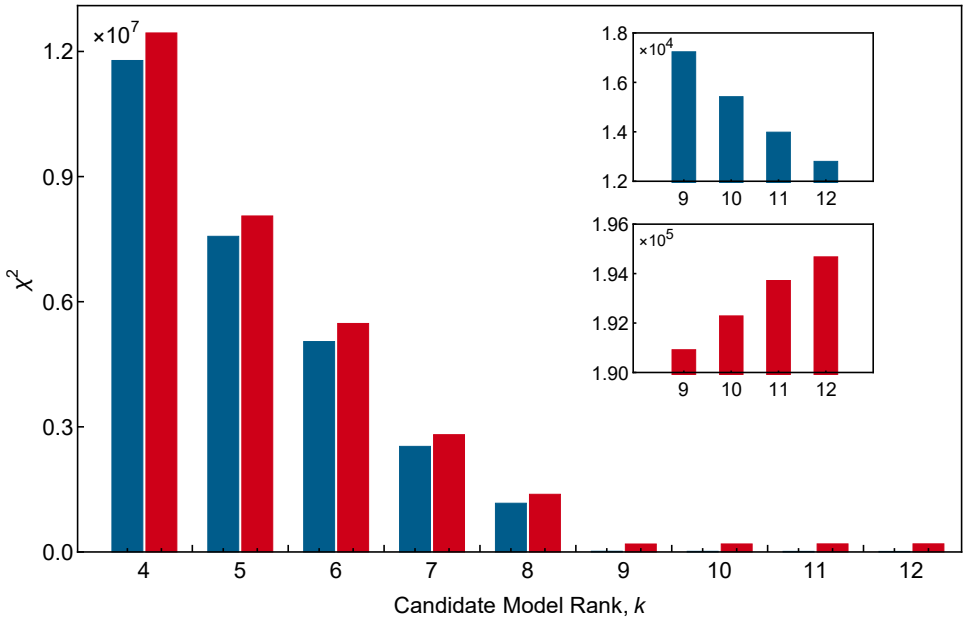

Figure 4.6: Training (blue) and testing (red) errors for the $m = n = 100$ Haar-distributed experiment corresponding to various candidate model ranks k. Insets show training and testing error for model ranks 9 through 12. Models with rank $k \leq 8$ severely under-fit the experimental data as indicated by the relatively high training error. Models with ranks $k > 9$ begin to over-fit the experimental data, as indicated by the increase in the testing error. We conclude that the best fit GPT corresponding to our data is rank $k = 9$ as it has the highest predictive power.

model underfits the data relative to the rank-k model, so that it has a higher training error and also a higher testing error, then every model with rank *less than* $k-1$ can only further underfit the data relative to the rank-k model. This means that every model with rank less than $k-1$ must have training and testing errors that are higher than or equal to those of the rank-$(k-1)$ model. On the other hand, if the rank-k model *overfits* the data relative to the rank-$(k-1)$ model, so that it has a lower training error but a higher testing error, then every model with rank *greater than* k can only further overfit the data relative to the rank-k model. This implies that the training error associated with models with rank greater than k can only fall further and their testing errors tend to rise higher. The change in the rate of decrease of the training error after the point where the testing error is minimized is also expected. Before that point, an increase in rank makes a big difference to the ability of the model to fit the data. After that point, the additional parametric

freedom can only cause the model to fit to small statistical fluctuations in the training data, which can only yield a slightly lower training error.

To summarize the above discussion, the trend of the training and test errors that we observed in Fig. 4.6 is precisely what one expects under the assumption of the correctness of quantum theory. That is, the rank $k = 9$ model is the one at which there is a transition from underfitting to overfitting. We therefore draw the conclusion that the true model underlying our data is most likely 9-dimensional, and we adopt D_9^{realized} as the best-fit probability matrix for our data.

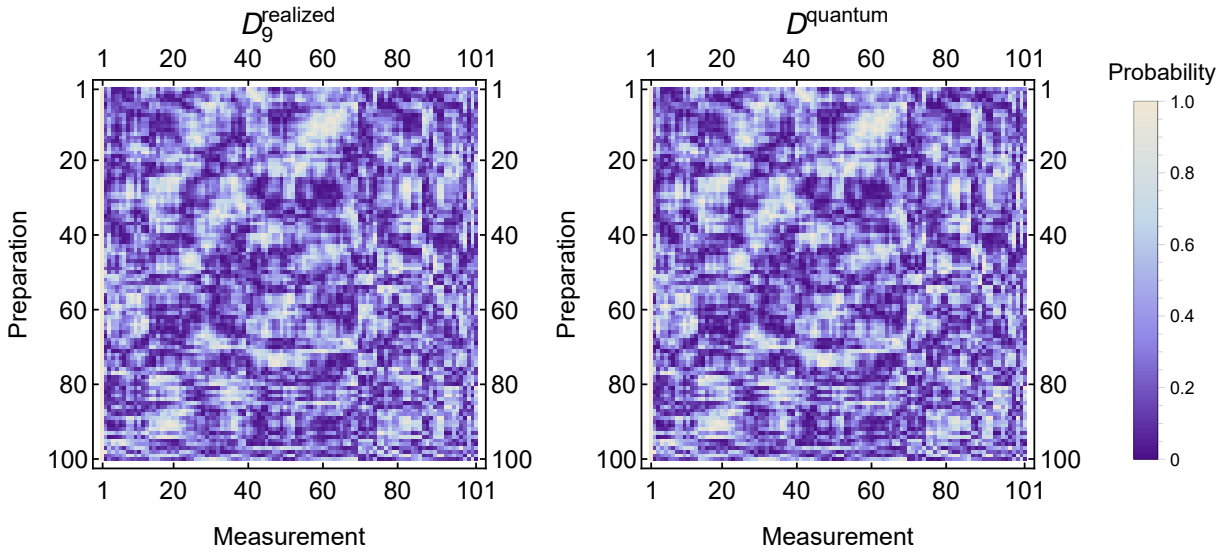

Figure 4.7: Comparison between the matrix plots of the best fit realized probability matrices D_9^{realized} and the predicted quantum mechanical probability matrices D^{quantum} for the $m = n = 100$ experiment. The mean and standard deviation of the element-wise difference between D_9^{realized} and D_9^{quantum} is -0.003 and 0.03, respectively.

From the above analysis, we identified that the best-fit probability matrix for our experimental data is the matrix D_9^{realized}. Figure 4.7 compares D_9^{realized} with the theoretical probability matrix predicted by qutrit quantum mechanics, denoted D^{quantum}. The quantum probabilities were computed using $D_{ij}^{\text{quantum}} = \text{Tr}[\rho_i Q_j]$, with ρ_i and Q_j being the same states and effects used to generate the wave plate configurations described in section 4.2.4. Both D_9^{realized} and D^{quantum} are displayed as matrix plots of probability values. The perceived messiness of the pattern in the matrix plots is a consequence of the random Haar-distributed experimental configurations. Figure 4.7 shows a large degree of agree-

ment between the experimentally realized probability matrix and the probability matrix predicted by quantum theory. Specifically, the mean and standard deviation of the difference between D_{ij}^{realized} and D_{ij}^{quantum} is calculated to be -0.003 and 0.03, respectively.

4.3.2 Increasing the Number of Experimental Configurations

The main result obtained from the $m = n = 100$ experiment was that the dimension of the GPT underlying our experiment is 9. In this section, we outline a method for increasing the number of GPT states and effects characterized by our experiment, so that we can gain a clearer picture of the geometric features of the experimentally realized GPT state and effect spaces.

Directly increasing m and n causes a large increase in the number of experimental configurations, and consequently, a drastic increase in the experimental run-time. However, to characterize a particular GPT state vector, we only need to find its statistics on any tomographically complete subset of measurements. Similarly, to characterize any GPT effect vector, we only need to implement it on a tomographically complete set of preparations. The result is that we should be able to characterize m preparations and n measurements with only $k(m + n)$ experimental settings rather than mn settings, where k is the rank of the model under consideration.

The first experiment was successful in identifying the rank of the GPT describing our data. Nevertheless, it is important to not assume this result when conducting our second experiment. Although it is likely that we only need 9 preparations/measurements to construct our tomographically complete sets, we will use more than that to account for the chance that our new data contradicts the rank-9 conclusion of our first experiment. Our choice for our tomographically complete set of states is the 15 unique normalized eigenvectors of the Gell-Mann matrices (excluding the identity matrix). For consistency in our experimental methods, our tomographically complete set of measurements are simply chosen as $\{Q, \bar{Q}\} = \{|\psi_i\rangle\langle\psi_i|, I - |\psi_i\rangle\langle\psi_i|\}$ for each normalized Gell-Mann matrix eigenvector $|\psi_i\rangle$. We call our sets of 15 tomographically complete states and effects our *fiducial* sets and we use them to increase the number of states and effects we can characterize with our experiment. To do this, we set $m = n = 400$ and generate our preparations and measurements according to the Haar-distribution (in the same manner as our first experiment). Each of the 15 fiducial states is than paired with the 400 Haar-distributed effects, and each of the 400 Harr-distributed states is paired with the 15 fiducial effects. Allowing each fiducial state to be paired with each fiducial effect increases the total number of preparations and measurements to $m = n = 415$ and adds 15^2 experimental settings,

for a total of $15(400 + 400 + 15) = 12225$. Thus, we are able to characterize more than four times the GPT states and effects than in our first experiment with only a 22% increase in experimental configurations. This ensures that the experimental run time is not drastically larger than in our first experiment. The unit effect $\boldsymbol{u}$ is also added to our second experiment so that our data will be arranged into a 415×416 frequency matrix F, where the bottom right block of 400×400 entries are unfilled (they do not correspond to preparation/measurement pairs actually measured in experiment). The structure of the F matrix for the fiducial experiment is shown in Figure 4.8 for clarity.

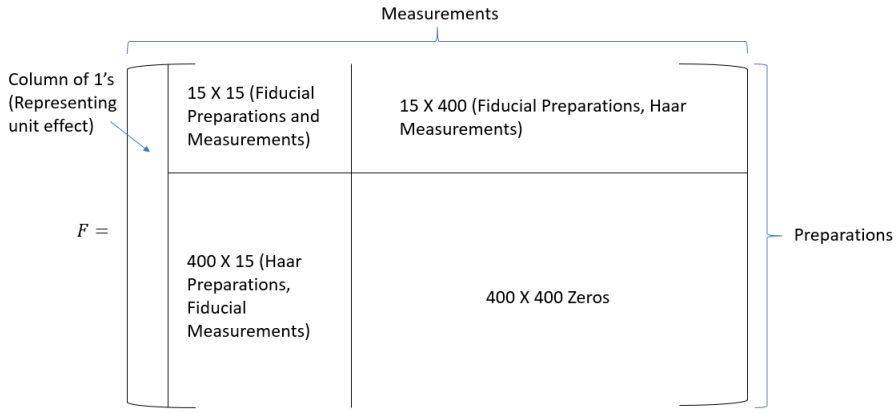

Figure 4.8: Format of the experimental frequency matrices for the fiducial experiment.

We run our second experiment twice back to back to obtain a training and test data set and perform the same analysis to that of our first experiment. We choose to only analyze the candidate model ranks $k \in \{8, 9, 10\}$ as our first experiment indicates that $k = 9$ is likely to provide the best fit for our data. Furthermore, as we explained above for the first experiment, if $k = 9$ is found to be the transition point between underfitting and overfitting in the range $k \in \{8, 9, 10\}$, then one can infer that ranks $k < 8$ could only further underfit the data while ranks $k > 10$ could only further overfit the data.

We find the best fit matrix D_k^{realized} for F^{train} using Eqs. (4.4) and (4.5), then retrieve the training error for each model by substituting D_k^{realized} into Eq. (4.1). The 400×400 sub-matrix of unfilled entries in the frequency matrices have uncertainty $\Delta F_{ij} = 0$. To avoid divergences to infinity arising from $1/\Delta F_{ij}$, we force $\Delta F_{ij} = \infty$ for these entries so that the weighting on the least squares fit is 0. Therefore, the only constraint on the fit for the 400×400 unfilled entries is that each corresponding entry in D_k^{realized} must be a valid probability. Once D_k^{realized} is found for each candidate model, we compute the corresponding testing errors using Eq. (4.6). The training and testing errors for the fiducial experiment

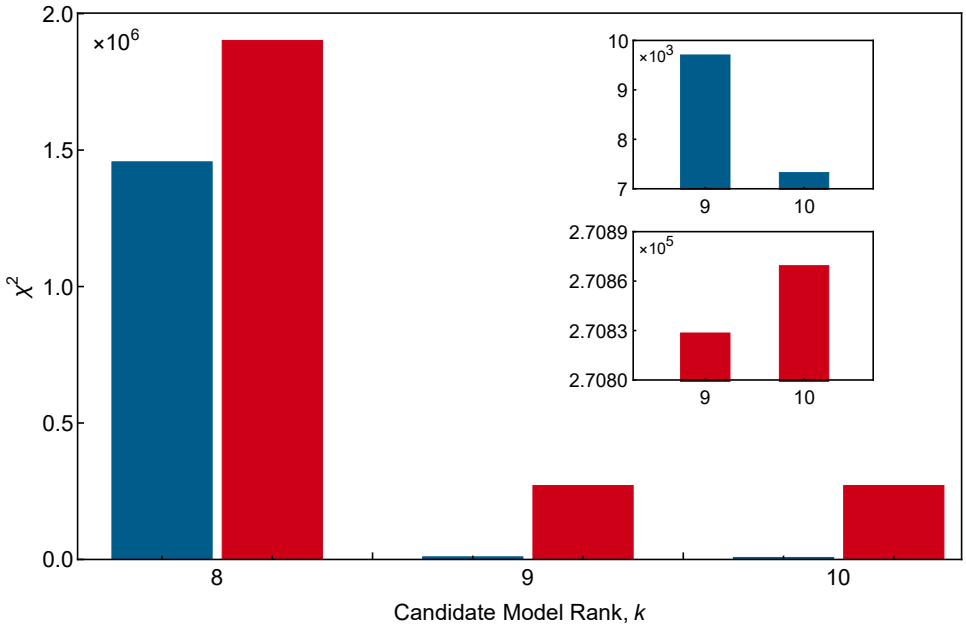

Figure 4.9: Training (blue) and testing (red) errors for the $m = n = 415$ fiducial experiment corresponding to various candidate model ranks k. The rank $k = 8$ model underfits the experimental data, while the rank $k = 10$ model overfits the data. We conclude that the best fit GPT corresponding to our data is rank $k = 9$ as it has the highest predictive power, in agreement with our first experiment.

were computed using only the elements of the frequency matrices actually measured. The unfilled 400×400 entries are excluded from this analysis as they would cause the training error to become large for all models.

The results of the training and testing error analysis for the fiducial experiment are shown in Figure 4.9. In agreement with the first experiment, the $k = 8$ model has relatively large training and testing errors indicating that it is under-fitting the experimental data. For the $k = 10$ model, the testing error indicates that the model is over-fitting the data. We draw the conclusion that in agreement with our first experiment, the model with the most predictive power for our data has rank $k = 9$ and we infer that D_9^{realized} is the best-fit probability matrix.

A comparison between D_9^{realized} for our fiducial experiment and the associated probability matrix predicted by qutrit quantum mechanics D^{quantum} is shown in figure 4.10. The narrow bands spanning the top and left sides of the matrix plots represent the prepara-

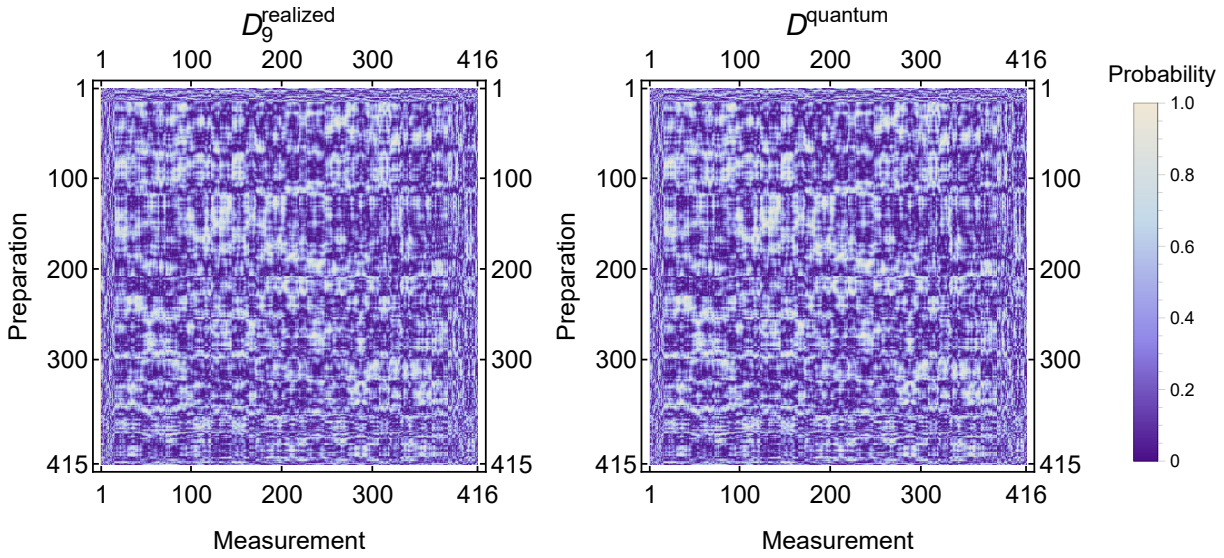

Figure 4.10: A comparison of the matrix plots of D_9^{realized} and D_9^{quantum} for the $m = n = 415$ fiducial experiment. The mean and standard deviation of the difference between D_9^{realized} and D_9^{quantum} is -0.006 and 0.03, respectively.

tion and measurement settings obtained in the experiment. These bands can be visually distinguished from the rest of the probabilities in the matrix plots because the 15 fiducial states (effects) are more distinct from one another than the 400 states (effects) randomly sampled from the Haar distribution. The remainder of the probabilities in D_9^{realized} are filled in from the weighted least squares fit, and were not recorded in the actual experiment. In agreement with our first experiment, we see a strong agreement between the experimental realized probability matrix and the probability matrix predicted by quantum theory with mean and standard deviation of the difference between D_{ij}^{realized} and D_{ij}^{quantum} being -0.006 and 0.03, respectively.

4.3.3 Decomposing D_9^{realized} into GPT State and Effect Vectors

We now discuss how to decompose the fiducial experiment D_9^{realized} into realized GPT state and effect vectors. Recall from section 3.2.1 that the decomposition $D = SE$ is not unique as any $k \times k$ invertible matrix A can be inserted into the decomposition such that $D = SAA^{-1}E$. It is expected that because the experimentally realized GPT has dimension 9, we will be able to recover close to the same sets of states and effects that we used to

generate the experimental preparations and measurements, and that there should exist a decomposition of D_9^{realized} that accomplishes this recovery. Therefore, the goal is to find a decomposition of D_9^{realized} that produces GPT state and effect vectors that are as close to the quantum state and effect vectors used to generate the experimental wave plate angles as possible. It is desirable to identify this specific decomposition as it will allow us to more easily compare our realized GPT spaces to those predicted by qutrit quantum theory.

To start, we find any valid decomposition of $D_9^{\text{realized}} = S'E'$ using the algorithm shown in appendix A.2.1. The algorithm utilizes the QR decomposition of an $m \times n$ matrix and proceeds as follows: We first perform the decomposition $D_9^{\text{realized}} = QR$, where Q is a 415×415 unitary matrix, and R is a 415×416 upper-triangular matrix. Since the first column of D_9^{realized} is filled with ones, the first column of Q must be filled with some constant a. By defining $Q' = Q/a$ and $R' = aR$, we can force the first column of Q' to be filled with ones. Following this, we let $Q' = (Q_0, Q_1)$, where Q_0 is the first column of Q' and Q_1 consists of the remaining columns of Q'. Similarly, we let $R' = (R_0, R_1)$, where R_0 is the first row of R' and R_1 consists of all other rows of R'. We then take the singular value decomposition $Q_1 R_1 = U\Sigma V^T$, where U is a 415×415 unitary matrix, Σ is the diagonal matrix containing the singular values of $Q_1 R_1$, and V is a 416×416 unitary matrix. Since $Q_1 R_1$ is rank 8, there are only 8 non-zero singular values. Thus, only the first 8 columns of U and V will contribute to $Q_1 R_1$. That is, $Q_1 R_1 = U_8 \Sigma_8 V_8^T$, where U_8 and V_8 consist of only the first 8 columns of U and V, and Σ_8 consists of the top left 8×8 corner of Σ. Finally, we define $S' = (Q_0 \;\; U_8\sqrt{\Sigma_8})$ and $E' = (R_0 \;\; \sqrt{\Sigma_8} V_8^T)^T$. Following this algorithm ensures that S' and E' are always of the form:

$$S' = \begin{pmatrix} 1 & s_1^{(1)} & \dots & s_8^{(1)} \\ 1 & s_1^{(2)} & \dots & s_8^{(2)} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & s_1^{(415)} & \dots & s_8^{(415)} \end{pmatrix} \quad \text{and} \quad E' = \begin{pmatrix} 1 & e_0^{(2)} & \dots & e_0^{(416)} \\ 0 & e_1^{(2)} & \dots & e_1^{(416)} \\ \vdots & \vdots & \vdots & \vdots \\ 0 & e_8^{(2)} & \dots & e_8^{(416)} \end{pmatrix}, \tag{4.7}$$

where $s_i^{(j)}$ is the i^{th} element of the GPT state vector corresponding to the j^{th} preparation, and $e_i^{(j)}$ is the i^{th} component of the GPT effect vector corresponding to outcome 0 of the j^{th} measurement.

We now show how the invertible matrix A that recovers GPT state and effect vectors that are as close to the quantum predictions as possible is computed from the decomposition $D_9^{\text{realized}} = S'E'$. Let S^{quantum} and E^{quantum} be the matrices that consist of the 415 and 416 Haar-distributed (and fiducial) qutrit quantum state and effect vectors used to generate the wave plate angles for the fiducial experiment. With this notation, the problem at hand

is to find a 9×9 invertible matrix A such that $S'A$ is as similar as possible to S^{quantum}. Therefore, we wish to find the matrix A that minimizes the least squares cost function

$$\begin{aligned} \text{Minimize} \quad & \chi^2 = \sum_i \sum_j (S_{ij}^{\text{quantum}} - (S^{'}A)_{ij})^2, \\ \text{Subject To:} \quad & A \quad \text{Invertible} \end{aligned} \tag{4.8}$$

where $i \in \{1, 2, \ldots, 415\}$ and $j \in \{1, 2, \ldots, 9\}$.

We identify the optimal transformation matrix using the Mathematica code shown in appendix A.3. Once the optimal transformation matrix, denoted $A_\star$, is computed from Eq. (4.8), we define $S^{\text{realized}} = S^{'}A_\star$ and $E^{\text{realized}} = A_\star^{-1}E^{'}$ to be the matrices consisting of our experimentally realized GPT state and effect vectors. Notice that by solving Eq. (4.8), we are not changing the values of the matrix of probabilities D_9^{realized}, and so the resulting decomposition $D_9^{\text{realized}} = S^{\text{realized}}E^{\text{realized}}$ does not require any additional assumptions regarding the correctness of quantum theory when compared with the initial decomposition $D_9^{\text{realized}} = S^{'}E^{'}$.

4.3.4 Geometry of the Experimentally Realized GPT

The experimentally realized GPT state space $\boldsymbol{\mathcal{S}}^{\text{realized}}$ is defined to be the convex hull of the rows of S^{realized}. The definition of the experimentally realized GPT effect space, denoted $\boldsymbol{\mathcal{E}}^{\text{realized}}$, is slightly more complicated. The realized effect vectors found from the columns of E^{realized}, and which correspond to outcome 0 of our experiment, can be written as the complements of the effects corresponding to the union of outcomes 1 and 2 in our experiment. That is, $\bar{\boldsymbol{e}}^{\text{realized}} = \boldsymbol{u} - \boldsymbol{e}^{\text{realized}}$. Thus, we define the realized effect space $\boldsymbol{\mathcal{E}}^{\text{realized}}$ as the convex hull of all columns of the matrix E^{realized} in union with their respective complements. Since the unit effect $\boldsymbol{u}$ is included in E^{realized}, the zero effect $\boldsymbol{0}$ is necessary included in $\boldsymbol{\mathcal{E}}^{\text{realized}}$ (as it is the complement of $\boldsymbol{u}$). Since there are 416 columns in E^{realized}, the convex hull representing $\boldsymbol{\mathcal{E}}^{\text{realized}}$ is taken over a total of 832 GPT effect vectors.

If the realized GPT spaces resemble the ones predicted by qutrit quantum theory, then we expect that their shapes should satisfy similar constraints to equations (1.31) and (1.39) that produced the projections shown in figure 1.2. Due to the convexity of the sets of preparations and measurements in our experiment, the realized GPT spaces will be represented by convex polytopes. As long as a large enough number of preparation and measurement settings are sampled in our experiment, the geometry of these polytopes should be a reasonably good approximation to the true GPT spaces governing our photonic

three-level system. Due to unavoidable noise in the implementation of the experiment, as well as the fact that we cannot sample infinite preparations and measurements, the realized GPT spaces will necessarily be strictly contained within the true GPT state and effect spaces describing our three-level system, in accordance with the discussion in section 3.3.

In figure 4.11, we display various three-dimensional projections of our experimentally realized GPT state and effect spaces. The projections of $\boldsymbol{\mathcal{S}}^{\text{realized}}$ are displayed as yellow polytopes. The projections shown in figure 4.11 match the quantum mechanical projections featured in figure 1.2. Similarly, projections of $\boldsymbol{\mathcal{E}}^{\text{realized}}$ are displayed as cyan polytopes. Again we choose projections that match those shown in figure 1.2. The projections displayed in figure 4.11 are in close agreement to what one would expect to see if quantum theory was indeed the correct representation of nature.

Recall from section 3.3 that $\boldsymbol{\mathcal{E}}^{\text{consistent}} = \text{dual}(\boldsymbol{\mathcal{S}}^{\text{realized}})$ and $\boldsymbol{\mathcal{S}}^{\text{consistent}} = \text{dual}(\boldsymbol{\mathcal{E}}^{\text{realized}})$. We can explicitly write out the definition of $\boldsymbol{\mathcal{E}}^{\text{consistent}}$ as the set of all vectors $\boldsymbol{e}$ such that $\forall \boldsymbol{s} \in \boldsymbol{\mathcal{S}}^{\text{realized}} : 0 \leq \boldsymbol{s} \cdot \boldsymbol{e} \leq 1$. Analogously, We can explicitly write out the definition of $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ as the set of all vectors $\boldsymbol{s}$ such that $\forall \boldsymbol{e} \in \boldsymbol{\mathcal{E}}^{\text{realized}} : 0 \leq \boldsymbol{s} \cdot \boldsymbol{e} \leq 1$ and $\boldsymbol{s} \cdot \boldsymbol{u} = 1$. The above definitions of $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ and $\boldsymbol{\mathcal{E}}^{\text{consistent}}$ are called the *inequality representations* of the spaces. However, in order to plot $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ and $\boldsymbol{\mathcal{E}}^{\text{consistent}}$ in the same way as $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{E}}^{\text{realized}}$, we require the *vertex representation* of the spaces, which lists all the vertices that specify their convex hulls. Computing a vertex representation from an inequality representation requires solving the *vertex enumeration problem*. The solution to the vertex enumeration problem is obtained using an algorithm first developed by Avis and Fukada [4]. Calculation of the vertex representations of our consistent spaces was performed using the lrs library in C that implements a modified version of the vertex enumeration algorithm [17].

The realized GPT spaces are necessarily smaller than the true GPT spaces, which implies that the consistent GPT spaces are necessarily larger than the logical GPT spaces (see section 3.3). With these set inclusion relations, we can infer that the true GPT state and effect spaces governing our three-level system must exist somewhere between the experimentally realized and the logically consistent spaces. The gap between our realized and consistent spaces allows us to infer a range of GPTs that could be used to describe our experimental data. A small gap between the two spaces is an indication that there are a limited number of GPTs that can plausibly describe the experimental data, which provides a qualitative bound on potential deviations from quantum theory.

Three-dimensional projections of the spaces $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ and $\boldsymbol{\mathcal{E}}^{\text{consistent}}$ are plotted as mesh polytopes alongside the same projections of $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{E}}^{\text{realized}}$ in figure 4.11. For the two-dimensional state space projections in figures 4.11(g)-(j), the consistent spaces are plotted in dark orange rather than as a mesh. The gaps between the realized and consistent spaces

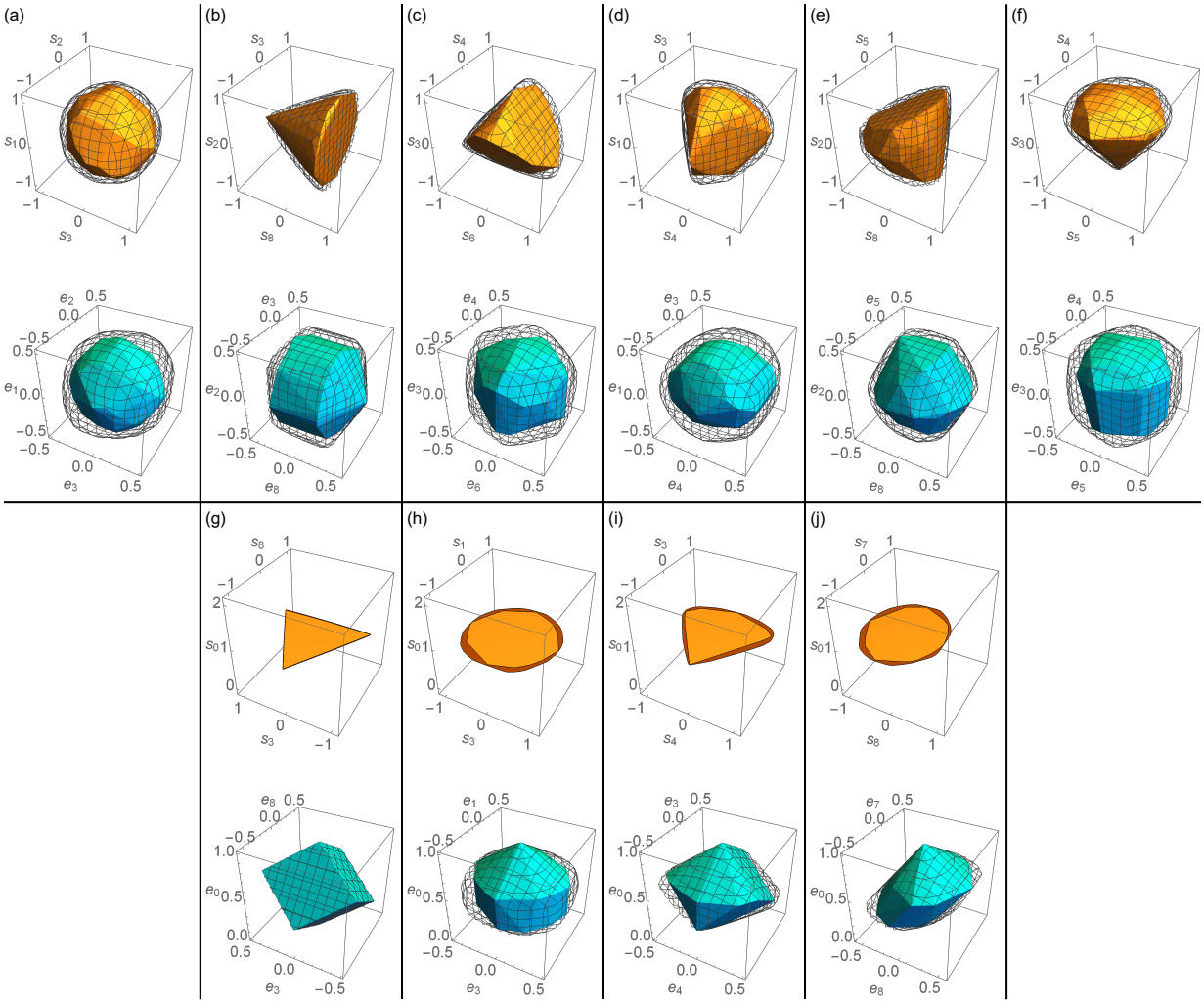

Figure 4.11: Three-dimensional projections of $\boldsymbol{\mathcal{S}}^{\text{realized}}$ (yellow polytopes) and $\boldsymbol{\mathcal{E}}^{\text{realized}}$ (cyan polytopes). The mesh polytopes surrounding each projection of $\boldsymbol{\mathcal{S}}^{\text{realized}}$ represent the analogous projections of $\boldsymbol{\mathcal{S}}^{\text{consistent}}$, which are calculated from the realized GPT effects. Similarly, the mesh polytopes surrounding each projection of $\boldsymbol{\mathcal{E}}^{\text{realized}}$ represent the analogous projections of $\boldsymbol{\mathcal{E}}^{\text{consistent}}$, which are calculated from the realized GPT states. (g)-(j) Projections involving the e_0 Bloch coefficient. Two-dimensional projections of $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ are shown in dark orange surrounding the corresponding projection of $\boldsymbol{\mathcal{S}}^{\text{realized}}$.

are not constant between different projections for both the state and effect spaces. This is due to our choices of preparation and measurement settings. Since our preparations and

measurements are drawn from a uniform Haar-distribution, some projections will be more densely sampled than others due to randomness.

4.3.5 Bounding Deviations from the No-Restriction Hypothesis

The gap between $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ provides a sense of the range of possible deviations from quantum theory. Specifically, they bound how far any candidate GPT describing our three-level system may deviate from the no-restriction hypothesis. We quantify this bound by identifying the points of intersection between a set of rays and our 8-dimensional realized and consistent GPT state spaces. This section describes the method of computing these points of intersection.

Each ray is chosen such that it lies along the direction of a random Haar-sampled pure qutrit quantum state. For each random pure state vector that lies along the direction of a ray, we shrink the vector such that it is guaranteed to lie inside $\boldsymbol{\mathcal{S}}^{\text{realized}}$, and denote this shortened vector as $\boldsymbol{s_1}$. In addition, we extend the pure state vector such that it is guaranteed to lie outside of $\boldsymbol{\mathcal{S}}^{\text{realized}}$, and denote this lengthened vector as $\boldsymbol{l_1}$. We then take the midpoint of $\boldsymbol{s_1}$ and $\boldsymbol{l_1}$, denoted $\boldsymbol{m_1}$, and check whether it lies inside or outside the polytope under consideration. If the midpoint is outside the polytope, then we set $\boldsymbol{l_2} = \boldsymbol{m_1}$. If the midpoint is inside the polytope, then we set $\boldsymbol{s_2} = \boldsymbol{m_1}$. Additional midpoints are computed and checked until $||\boldsymbol{l_i}|| - ||\boldsymbol{s_j}|| < 10^{-4}$, and we take the final midpoint $\boldsymbol{p} = (\boldsymbol{l_i} + \boldsymbol{s_j})/2$ to be the point of intersection between the ray and the polytope. This process is repeated for $\boldsymbol{\mathcal{S}}^{\text{consistent}}$.

An example illustrates the algorithm. Consider the two-dimensional convex polytope shown in Fig. 4.12. A ray of random direction is represented as a blue line. We wish to find the intersection point between this ray and the polytope. In Fig. 4.12(a), we begin with the points $\boldsymbol{s_1}$ and $\boldsymbol{l_1}$ which lie inside and outside the polytope, respectively. We observe that midpoint $\boldsymbol{m_1}$ lies outside the polytope. Thus, we set $\boldsymbol{l_2} = \boldsymbol{m_1}$. In Fig. 4.12(b), we take the midpoint $\boldsymbol{m_2}$ between $\boldsymbol{s_1}$ and $\boldsymbol{l_2}$ and observe that it lies inside the polytope. Therefore, we set $\boldsymbol{s_2} = \boldsymbol{m_2}$. In Fig. 4.12(c), the midpoint $\boldsymbol{m_3}$ between $\boldsymbol{s_2}$ and $\boldsymbol{l_2}$ is taken. It is clear that if this process is repeated, we will converge on the point of intersection with the polytope.

To determine whether a given point lies inside our outside a convex polytope, one needs to check if there exists a convex combination of vertices that yield the point of interest. If a convex combination exists, then the point is inside the polytope. Otherwise, the point lies outside the polytope. We check for the existence of a convex combination of vertices that yield a point of interest $\boldsymbol{b}$ by organizing the vertices of the polytope into columns of

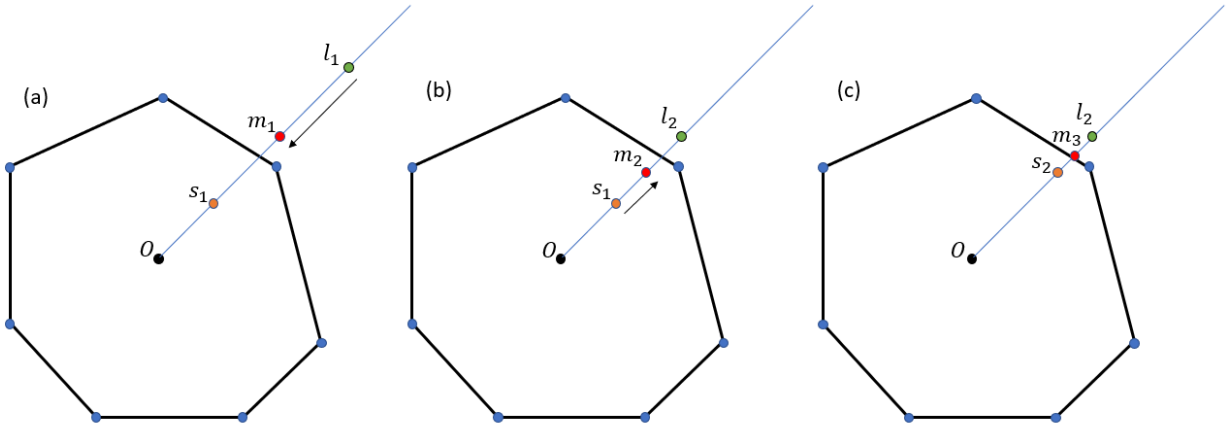

Figure 4.12: Example illustrating the algorithm for finding the point of intersection of a random ray and a convex polytope.

a matrix V. We then check for the existence of a vector of weights $\boldsymbol{x}$ such that $V\boldsymbol{x} = \boldsymbol{b}$, subject to $0 \leq x_i \leq 1$.

The Mathematica code used to find the points of intersection between random rays and our GPT spaces $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ is shown in appendix A.4. For each ray, we divide the norm of the intersection point with $\boldsymbol{\mathcal{S}}^{\text{realized}}$ by the norm of the intersection point with $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ to obtain a *linear dimension ratio*. For a set of 1000 rays, the average linear dimension ratio between $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ is computed to be 0.80, with an associated standard deviation of 0.04. The average linear dimension ratio obtained from $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ provides a quantitative lower bound on the extent to which the no-restriction hypothesis can be violated by GPTs that aim to describe our three-level system. To be specific, any candidate GPT describing our three-level system that admits an average linear dimension ratio much lower than 0.80 deviates farther from the no-restriction hypothesis than our experimental data predicts is possible, and can be ruled out with confidence.

4.3.6 Consistency with Quantum Theory

To conclude this chapter, we discuss the consistency of our realized and consistent GPT state spaces with quantum theory. By the GPT inference problem, it is expected that the qutrit quantum state space $\boldsymbol{\mathcal{S}}^{\text{quantum}}$ must lie strictly between $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{S}}^{\text{consistent}}$, up to a linear transformation. However, noise in the implementation of an experiment may cause some realized state vectors to be longer than they should be. To elaborate, the rows

of the matrix S^{realized} are only finite-sample estimations of the states in $\boldsymbol{\mathcal{S}}^{\text{realized}}$. A given state vector in $\boldsymbol{\mathcal{S}}^{\text{realized}}$ specifies the long-run probabilities that would be obtained by the associated realized preparation procedure, and the finite-run relative frequencies obtained in our experiment will not necessarily agree with these long-run probabilities. Therefore, finite-run estimates of the realized state vectors will deviate away from the realized state vectors, and may be longer than they otherwise would be. In an analogous way, finite-run estimates of certain realized effect vectors might be abnormally long, implying that the corresponding finite-run estimates of certain consistent state vectors might be shorter than they otherwise would be. The conclusion is that the finite sampling in the experiment can lead to a situation where $\boldsymbol{\mathcal{S}}^{\text{qutrit}}$ does not fit between our estimates of $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{S}}^{\text{consistent}}$.

Due to the fact that the geometry of the qutrit state space is high-dimensional, and that our estimate of the consistent state space has a very large number of vertices, it is challenging to determine if a linear transformation exists in which $\boldsymbol{\mathcal{S}}^{\text{qutrit}}$ fits between $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{S}}^{\text{consistent}}$. We therefore resort to a heuristic approach to answering this question. We again consider the 1000 rays sampled according to the Haar measure and with directions defined by pure qutrit quantum states from section 4.3.5. Let l be an index that runs over these rays. For each l, $\boldsymbol{s}_l^{\text{realized}}$ is defined to be the vector in the direction of the lth ray that lies on the boundary of $\boldsymbol{\mathcal{S}}^{\text{realized}}$. Similarly, $\boldsymbol{s}_l^{\text{consistent}}$ is defined to be the vector in the direction of the lth ray that lies on the boundary of $\boldsymbol{\mathcal{S}}^{\text{consistent}}$. Recall from Eq. (1.31) that all pure qutrit states have Bloch vectors with the same norm (namely $2/\sqrt{3}$). Therefore, as long as the longest of the $\boldsymbol{s}_l^{\text{realized}}$ vectors is shorter than the shortest of the $\boldsymbol{s}_l^{\text{consistent}}$, that is, if $\max_l ||\boldsymbol{s}_l^{\text{realized}}|| \leq \min_l ||\boldsymbol{s}_l^{\text{consistent}}||$, then at least along the 1000 sampled random directions, there must be a linear transformation of $\boldsymbol{\mathcal{S}}^{\text{qutrit}}$ which is straddled by our estimates of $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{S}}^{\text{consistent}}$. In practice, we find $\max_l ||\boldsymbol{s}_l^{\text{realized}}|| = 1.11$ and $\min_l ||\boldsymbol{s}_l^{\text{consistent}}|| = 1.13$, so the straddling condition does hold for the sampled 1000 random directions. To conclude, our experimental results obtained from the 1000 sampled directions provides evidence in favour of the hypothesis that quantum theory is the true GPT underlying our experiment.

Chapter 5

Conclusion

In this work, we reconstructed the GPT state and effect spaces for a three-level photonic system directly from experimental data. The tomographic scheme that we used in the experiment was self-consistent in that it did not require any prior representations of the preparations and measurements in advance. Furthermore, our experimental scheme as well as our analysis techniques avoided assuming the correctness of quantum theory. From the statistics obtained in our experiment, we were able to infer that the most likely dimension of the underlying GPT for our photonic system is 9. This was accomplished through the computation of the training and testing errors (Eqs (4.1) and (4.6)) for a variety of best-fit probability matrices of differing ranks. This model selection technique provided an opportunity to discover that the dimension of the GPT governing our three-level system differed from the quantum mechanical predictions. However, none of our analysis yielded any evidence in favour of such a deviation.

Following the dimensionality analysis, we identified a range of possible geometries for GPTs that are consistent with our experimental data. The true GPT state and effect spaces describing our three-level system must lie somewhere between our experimentally realized GPT spaces ($\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{E}}^{\text{realized}}$) and the logically consistent GPT spaces ($\boldsymbol{\mathcal{S}}^{\text{consistent}}$ and $\boldsymbol{\mathcal{E}}^{\text{consistent}}$). We displayed the geometries of our experimentally realized and logically consistent state and effect spaces as three-dimensional projections of the full nine-dimensional spaces in Fig. 4.11. In addition to the visualization of our realized and consistent GPT vector spaces, we were able to place quantitative bounds on how much any proposed alternative to quantum theory may violate the no-restriction hypothesis. We accomplished this by identifying the average linear dimension ratio between $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{S}}^{\text{consistent}}$ to be 0.80. Even though our experimental data provides some allowances for violations of the no-restriction hypothesis, we observe close agreement in the shapes of our experimental

GPT spaces when comparing them to the quantum mechanical predictions for a qutrit. We draw the conclusion that any three-level GPT having the potential to provide a more concrete description of nature of three-level systems must share many of the same geometric features as quantum theory.

There are many novel experiments of this kind that can be explored in future work. One obvious extension would be to find a GPT characterization for higher-dimensional systems, or for composite systems. As an example, one may attempt to experimentally construct a GPT characterization of an entangled pair of qubits and test possible deviations from properties that govern such systems. Another interesting potential for further work is to explore experimental GPT characterizations of channels for a given type of system by conducting the GPT equivalent of quantum process tomography. The results of our analysis and that of Ref. [34] for two-level systems demonstrate the incredible versatility of the GPT framework in testing the validity of potential alternatives to quantum theory within the full landscape of possible physical theories.

References

[1] Stephen L. Adler. Generalized quantum dynamics. *Nuclear Physics B*, 415(1):195–242, 1994.

[2] H. Akaike. *Information Theory and an Extension of the Maximum Likelihood Principle.* Springer, New York, 1998.

[3] Martín Avendaño. Descartes' rule of signs is exact! *Journal of Algebra*, 324(10):2884–2892, 2010.

[4] David Avis and Komei Fukuda. A pivoting algorithm for convex hulls and vertex enumeration of arrangements and polyhedra. *Proceedings of the Annual Symposium on Computational Geometry*, 313:98–104, 1991.

[5] Jonathan Barrett. Information processing in generalized probabilistic theories. *Physical Review A - Atomic, Molecular, and Optical Physics*, 75(3), mar 2007.

[6] E.T. Bell. Exponential polynomials. *Annals of Mathematics*, 35(2):258–277, 1934.

[7] Ingemar Bengtsson, Stephan Weis, and Karol Życzkowski. Geometry of the set of mixed quantum states: An apophatic approach. In Piotr Kielanowski, S. Twareque Ali, Anatol Odzijewicz, Martin Schlichenmaier, and Theodore Voronov, editors, *Geometric Methods in Physics*, pages 175–197, Basel, 2013. Springer Basel.

[8] Ingemar Bengtsson and Karol Życzkowski. *The space of density matrices.*, pages 209–232. Cambridge University Press, 2006.

[9] Robert W. Boyd. *Nonlinear Optics, Third Edition.* Academic Press, Inc., USA, 3rd edition, 2008.

[10] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[11] Stephen Boyd and Lieven Vandenberghe. *Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares*. Cambridge University Press, 2018.

[12] E. Brüning, H. Mäkelä, A. Messina, and F. Petruccione. Parametrizations of density matrices. *Journal of Modern Optics*, 59(1):1–20, 2012.

[13] Giulio Chiribella, Giacomo Mauro D'Ariano, and Paolo Perinotti. Probabilistic theories with purification. *Physical Review A - Atomic, Molecular, and Optical Physics*, 81(6), jun 2010.

[14] Giulio Chiribella, Giacomo Mauro D'Ariano, and Paolo Perinotti. Probabilistic theories with purification. *Phys. Rev. A*, 81:062348, Jun 2010.

[15] Giulio Chiribella, Giacomo Mauro D'Ariano, and Paolo Perinotti. Informational derivation of quantum theory. *Physical Review A - Atomic, Molecular, and Optical Physics*, 84(1), jul 2011.

[16] J.B. Conway. *A Course in Functional Analysis*. Springer-Verlag New York, 2007.

[17] D. Avis. lrs home page. http://cgm.cs.mcgill.ca/~avis/C/lrs.html, 2021.

[18] Giacomo Mauro D'Ariano. *Probabilistic theories: What is special about Quantum Mechanics?*, page 85–126. Cambridge University Press, 2010.

[19] G. C. Ghirardi, A. Rimini, and T. Weber. Unified dynamics for microscopic and macroscopic systems. *Phys. Rev. D*, 34:470–491, Jul 1986.

[20] Sandeep K Goyal, B Neethi Simon, Rajeev Singh, and Sudhavathani Simon. Geometry of the generalized bloch sphere for qutrits. *Journal of Physics A: Mathematical and Theoretical*, 49(16):165203, mar 2016.

[21] Nathan Hagen, Prathan Buranasiri, and Yukitoshi Otani. Alignment precision of polarization components. *Applied Optics*, 58(36):9750, 2019.

[22] Lucien Hardy. Quantum Theory From Five Reasonable Axioms. jan 2001.

[23] Lucien Hardy. Foliable operational structures for general probabilistic theories. In *Deep Beauty: Understanding the Quantum World Through Mathematical Innovation*, pages 409–442. Cambridge University Press, jan 2011.

[24] E. Hecht. *Optics*. Pearson education, Addison-Wesley, 2002.

[25] C. K. Hong and L. Mandel. Theory of parametric frequency down conversion of light. *Phys. Rev. A*, 31:2409–2418, Apr 1985.

[26] Peter Janotta and Haye Hinrichsen. Generalized probability theories: what determines the structure of quantum theory? *J. Phys. A: Math. Theor.*, 47(32):323001, jul 2014.

[27] Thomas Kauten, Robert Keil, Thomas Kaufmann, Benedikt Pressl, Časlav Brukner, and Gregor Weihs. Obtaining tight bounds on higher-order interferences with a 5-path interferometer. *New Journal of Physics*, 19(3):033017, mar 2017.

[28] Gen Kimura. The bloch vector for n-level systems. *Physics Letters A*, 314(5):339–349, 2003.

[29] Ulf Leonhardt. *Essential Quantum Optics: From Quantum Measurements to Black Holes*. Cambridge University Press, 2010.

[30] Hoi Kwong Lo and Sandu Popescu. Concentrating entanglement by local actions: Beyond mean values. *Physical Review A - Atomic, Molecular, and Optical Physics*, 63(2):1–16, 2001.

[31] R. Loudon. *The Quantum Theory of Light*. Clarendon Press, Oxford, 1983.

[32] Alexander Lvovsky. *Quantum Physics: An Introduction Based on Photons*. Springer-Verlag Berlin Heidelberg, 2018.

[33] Lluís Masanes and Markus P Müller. A derivation of quantum theory from physical requirements. *New J. Phys.*, 13(6):063001, jun 2011.

[34] Michael D. Mazurek, Matthew F. Pusey, Kevin J. Resch, and Robert W. Spekkens. Experimentally bounding deviations from quantum theory in the landscape of generalized probabilistic theories. *PRX Quantum*, 2:020302, Apr 2021.

[35] Elizabeth S. Meckes. *Haar Measure on the Classical Compact Matrix Groups*, page 1–30. Cambridge Tracts in Mathematics. Cambridge University Press, 2019.

[36] Francesco Mezzadri. How to generate random matrices from the classical compact groups. *Notices of the American Mathematical Society*, 54, 10 2006.

[37] G. J. Milburn. Intrinsic decoherence in quantum mechanics. *Phys. Rev. A*, 44:5401–5406, Nov 1991.

[38] Michal Mičuda, Ester Doláková, Ivo Straka, Martina Miková, Miloslav Dušek, Jaromír Fiurášek, and Miroslav Ježek. Highly stable polarization independent mach-zehnder interferometer. *Review of Scientific Instruments*, 85(8):083103, 2014.

[39] Hoban M. et al. Navascués M., Guryanova Y. Almost quantum correlations. *Nat Commun*, 6:6288, Nov 2015.

[40] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010.

[41] D K Park, O Moussa, and R Laflamme. Three path interference using nuclear magnetic resonance: a test of the consistency of born's rule. *New Journal of Physics*, 14(11):113025, nov 2012.

[42] Pennylane dev team. Understanding the haar measure. https://pennylane.ai/qml/demos/tutorial_haar_measure.html#:~:text=For%20an%20N%20%2Ddimensional%20system,its%20integral%20over%20the%20group., 2021.

[43] Ian C. Percival. *Environmental and Primary Quantum State Diffusion*. Springer, 1995.

[44] Asher Peres. Proposed test for complex versus quaternion quantum theory. *Phys. Rev. Lett.*, 42:683–686, Mar 1979.

[45] Terry Rudolph. Ontological Models for Quantum Mechanics and the Kochen-Specker theorem. 2006.

[46] Ana Belén Sainz, Yelena Guryanova, Antonio Acín, and Miguel Navascués. Almost-Quantum Correlations Violate the No-Restriction Hypothesis. *Physical Review Letters*, 120(20), may 2018.

[47] Ana Belén Sainz, Yelena Guryanova, Antonio Acín, and Miguel Navascués. Almost-quantum correlations violate the no-restriction hypothesis. *Phys. Rev. Lett.*, 120:200402, May 2018.

[48] Carlo Maria Scandolo. Information-theoretic foundations of thermodynamics in general probabilistic theories. jan 2019.

[49] Anthony J Short and Jonathan Barrett. Strong nonlocality: a trade-off between states and measurements. *New Journal of Physics*, 12(3):033034, mar 2010.

[50] Urbasi Sinha, Christophe Couteau, Thomas Jennewein, Raymond Laflamme, and Gregor Weihs. Ruling out multi-order interference in quantum mechanics. *Science*, 329(5990):418–421, 2010.

[51] Rafael D. Sorkin. Quantum mechanics as quantum measure theory. *Modern Physics Letters A*, 09(33):3119–3127, 1994.

[52] P. C. E. Stamp. Environmental decoherence versus intrinsic decoherence. *Phil. Trans. R. Soc. A.*, 370:4429, 2012.

[53] Stover, Christopher. Gell-mann matrix. https://mathworld.wolfram.com/Gell-MannMatrix.html.

[54] Guifré Vidal, Daniel Jonathan, and M. A. Nielsen. Approximate transformations and robust manipulation of bipartite pure-state entanglement. *Physical Review A - Atomic, Molecular, and Optical Physics*, 62(1):10, 2000.

[55] Mirjam Weilenmann and Roger Colbeck. Analysing causal structures in generalised probabilistic theories. *Quantum*, 4:236, February 2020.

[56] Weisstein, Eric W. Characteristic polynomial. https://mathworld.wolfram.com/CharacteristicPolynomial.html.

[57] Karol Zyczkowski. Geometry of quantum entanglement. https://pirsa.org/16120010, 2016.

APPENDICES

Appendix A

Selected Code

This appendix lists some selected code that was used throughout this thesis. Section A.1 contains the Mathematica algorithm for computing a set of Haar-distributed quantum states. Section A.2 contains the python 3 code for computing the probability matrices of best-fit using an alternating least squares algorithm. In addition, this section also contains the code for computing the initial decomposition of D^{realized}. Section A.3 contains the Mathematica code for identifying the optimal transformation matrix $A_\star$ such that $D = SA_\star A_\star^{-1}E$. Finally, section A.4 contains the Mathematica code for finding the points of intersection between a set of random rays and the spaces $\boldsymbol{\mathcal{S}}^{\text{realized}}$ and $\boldsymbol{\mathcal{S}}^{\text{consistent}}$.

A.1 Generating Haar-Distributed Quantum States

Mathematica code for generating a set of Haar-distributed quantum states of dimension d.

```
(*Author: Michael Grabowecky, QOQI 2021
enter the dimension of the system under consideration, qutrit: d = \
3*)
d = 3;
(*enter the number of Haar-distributed quantum states to generate*)
num = 1000000;
(*Loop for generating Haar-distributed states*)
densmat = Reap[Do[
```

```
     (*generate a complex matrix X with standard normal complex \
numbers as elements*)
     StartingMatrix =
      Table[Table[
        RandomVariate[NormalDistribution[0, 1]] +
         I*RandomVariate[NormalDistribution[0, 1]], {i, 1, d}], {j, 1,
         d}];
     (*perform the QR decomposition of X*)
     QRDecomp = QRDecomposition[StartingMatrix];
     HarrRandUnitary = QRDecomp[[1]];
     (*Compute matrix that makes QR decomp unique*)
     Lambda =
      DiagonalMatrix[
       Table[QRDecomp[[2]][[i, i]]/Abs[QRDecomp[[2]][[i, i]]], {i, 1,
         d}]];
     HarrRandUnitary = HarrRandUnitary.Lambda;
     (*retrieve pure states from columns of random unitary matrices*)
     haarpure = HarrRandUnitary[[All, 1]];
     haar = ArrayReshape[haarpure, {d, 1}];
     haarconj = ConjugateTranspose[haar];
     (*convert from state vectors to pure state density matrices*)
     haardens = haar.haarconj;
     Sow[haardens];
     i = i + 1, {i, 1, num}]][[2]];
```

A.2 Computing the Best-Fit Probability Matrix to the Experimental Data

Python 3 code for solving the best-fit alternating least squares optimization problem yielding D_k^{realized} (solution to Eq. (4.1)). This code was originally written by Tim Hill and was adapted for the purposes of this thesis.

```
"""
Author: Original script by Tim Hill, Adapted by Michael Grabowecky

Script for finding weighted low-rank approximation.
```

```
This script is the python equivalent to the series of Matlab scripts
    lowRankApproximation... .m

For a full-rank noisy probability matrix D, with Poisson uncertainties
U, we find the weighted low rank approximation which minimizes the chi-squared
statistic, using weights W = U^-2

---------------------------------------------------------------------
This version uses the cvxopt package (http://cvxopt.org/) to find the
optimal solution to each iteration's quadratic programming problem.

Documentation for cvxopt is here
    <http://cvxopt.org/userguide/>
Specifically for the quadratic program solver
    <http://cvxopt.org/userguide/coneprog.html#cvxopt.solvers.qp>

---------------------------------------------------------------------
I use the optional MOSEK solver, which lives at
<https://www.mosek.com/>

This is a commercial package, but has a free academic license.

User guide:
<http://docs.mosek.com/7.1/pythonapi/index.html>
---------------------------------------------------------------------
COMMAND LINE INTERFACE

quadsolve.py file rank [.. rank] [-f --fiducial]

Args:
 *  file:               Path to probabilities file
 *  rank:               What rank matrix to fit (1 or more)
 *  -t --tol            Chi-squared percentage tolerance

The file names for states, effects, uncertainties are automatically generated
from the file path to the probabilities.
---------------------------------------------------------------------
```

```
"""

import time
import os
import warnings
import argparse

import numpy as np
from scipy import sparse
import cvxopt
from cvxopt import msk
import mosek
from matplotlib import pyplot as plt

import decomp
import timing
import fantlk

# Turn off progress log for each qp call, and disable warnings
cvxopt.solvers.options['mosek'] = { mosek.iparam.log:0,
                                    mosek.iparam.max_num_warnings:0}


def vec(v):
    """Flattens the array v in column-major order.

    That is, for an m x n matrix v, the result is a (mn, ) size np.ndarray
    where the first m values are the first column of v, the m+1 to 2m values
    are the second column of v, and so on.

    This is the equivalent of the MATLAB operation M(:) on a matrix M.

    (This just simply wraps np.ndarray.flatten("F") but makes it more
    obvious what it's doing)
    """
    return v.flatten('F')

def toarray(v, shape):
```

```
    """Turns a flattened vector into a 2D array.

    This function undoes vec(v). That is, for any array A,
            toarray(vec(A)) == A    is element-wise True

    This function really just wraps np.reshape, using order = "F"
    (column-major order).
    """
    return v.reshape(shape, order = "F")

def cvxopt_qp(*args, **kwargs):
    """
    cvxopt_qp solves the quadratic problem given by *args using cvxopt.

    Function convers args to cvxopt.matrix type and passes args and kwargs
    to the cvxopt.solvers.qp function. This wrapper performs the type
    conversion on all args, and leaves kwargs alone.
    """
    return cvxopt.solvers.qp(*[cvxopt.matrix(arg) for arg in args], **kwargs)

def _raise_status(results):
    """Interpret the status for qp solution.

    Assuming the mosek solver was used, this function
    interprets the results dictionary, and raises ValueError
    for any non-optimal status code.

    Args:
     *  results: A results dictionary from the cvxopt.solvers.qp function,
            assuming solver = 'mosek'. ie,

        cvxopt.solvers.qp(P, q[, G, h[, A, b[, solver = 'mosek'[, initvals]]]])

    Returns: None

    Raises:
     *  ValueError for any status code other than "optimal"
     """
```

```
    _status_str = 'status'
    _pass = 'optimal'
    status = results[_status_str]
    if status != _pass:
        raise ValueError("Optimal solution not found. Status: %s" % status)

def low_rank_approx(d_file, k, fiducial = False, tol = 0.0001): #was 0.0001)
    """Finds a weighted low-rank approximation to d.

    This uses the cvxopt solver, which writes the problem in the form

                (1/2) x.T * P * x + q.T*x
    Subject to
            Gx <= h
            Ax = b  (we don't use this one)

    Parameters:
    *   d_file: path to file with simulated/experimental probabilities
    *   k: Rank of matrix to fit

    Returns:
     *  chi2: The chi-squared value the function converged to, or the final
            chi-squared that was calculated if the max number of
            iterations was reached before convergence
     *  status_code: Bool indicating whether the chi-squared converged.
            True: chi-squared converged to within tolerance
            False: Max iterations reached without converging
     *  pdiff: The percent-difference between last two chi-squared values
     *  i: The number of iterations used

     Raises:
      * ValueError: Optimal solution not found. Status {status code}
            For any status in the quadratic program solver other than
            "optimal". {status code} is replaced with the status code
            returned by the mosek solver, which will be one of
            * "primal infeasible"
            * "dual infeasible"
            * "unknown"
```

```
        See the cvxopt docs for an explanation of these codes
        <http://cvxopt.org/userguide/coneprog.html#cvxopt.solvers.qp>
"""
_maxiter = 1000
_start_time = time.time()

_pass = 'optimal'
_vec_str = 'x'
_break = 50 * "-"



if type(d_file) == str:
    print("Loading probabilities from %s" % d_file)
    d = np.loadtxt(d_file)
    fname, fext = os.path.splitext(d_file)
    file_fmt = "".join([fname + "_%s", fext])
    print("Outputting files using: %s" % file_fmt)
    w = np.loadtxt(file_fmt % "uncerts")
    dseed = np.loadtxt("trainfiducialFmatrix29012021_d_rank_8.txt")
    save = True
    s_recovery = file_fmt % "s_recovered"
    e_recovery = file_fmt % "e_recovered"

else:
    d, w = d_file
    save = False
    s_recovery = "s_recovered.txt"
    e_recovery = "e_recovered.txt"

print("Original matrix is rank %s" % np.linalg.matrix_rank(d))

w[w==0] = np.inf    # Set uncerainty to infinity so weight is 0
w = w**(-2)
w = sparse.diags(vec(w))

# Use full matrix decomposition for first guess
s, e = decomp.matrix_decomposition(d, k, False)
```

```
#s = np.loadtxt("trainfiducialFmatrix29012021_s_rank_9.txt")
#e = np.loadtxt("trainfiducialFmatrix29012021_e_rank_9.txt")
delta_d = sparse.lil_matrix(vec(d) - vec(np.dot(s, e))).T
chi2 = float((delta_d.T * w * delta_d.A))
print("Inital chi-square: %.5e" % chi2)
print("Starting iterative qp fit")
print(_break)
status_code = None

m, n = d.shape
l, r = s.shape
chi2_prev = np.inf

for i in range(1, _maxiter):
    if i % 100 == 0:
        print("Iteration = %s" % i)
        print("Total time: %s" % (time.time() - _start_time))

    ## ================================================================
    ##  Minimize S
    ## ================================================================
    try:
        # kronecker prod. of E, I(m)
        EI = sparse.lil_matrix(np.kron(e.T, np.eye(m)))

        # Q is the SPARSE quadratic cost matrix
        Q = EI.T * w * EI
        Q = 0.5 * (Q + Q.T)     # Ensure it's symmetric

        # q is the linear cost vector
        q = - (EI.T*w.dot(vec(d)))[m:] +
        np.hstack(np.dot(Q[:m, m:].A.T, np.ones((m, 1))))

        zcons = -EI[:, m:]          # Enforce   -D <= 0
        ocons =  EI[:, m:]          # Enforce    D <= 1
        # matrix_cons is the matrix G where Gx <= h
        matrix_cons = sparse.bmat([ [zcons], [ocons] ])
```

```
    n_0s = zcons.shape[0]       # Number of conditions -D <= 0
    n_1s = ocons.shape[0]       # Number of conditions  D <= 1
    # v_cons is the vector h where Gx <= h
    v_cons = np.concatenate(( np.zeros((n_0s, 1)) +
    EI[:, :m].dot(np.ones((m, 1))),
                              np.ones((n_1s, 1))
                              - EI[:, :m].dot(np.ones((m, 1)))))

    # results of quadratic optimization for the above values
    s_results = cvxopt_qp(Q.A[m:, m:], q, matrix_cons.A,
                                 v_cons.flatten(), solver = 'mosek')
    _raise_status(s_results)

    # s in vector form
    s_vec = s_results[_vec_str]
    # matrix-form
    s = toarray(np.concatenate((np.ones((m, 1)),
    np.array(s_vec))), s.shape)
except KeyboardInterrupt:
    print("Saving current iteration before quitting ...")
    np.savetxt(s_recovery, s)
    np.savetxt(e_recovery, e)
    raise

## ================================================================
##  Minimize E
## ================================================================
try:
    IS = sparse.lil_matrix(np.kron(np.eye(n), s))
    Q = IS.T * w * IS
    Q = 0.5 * (Q + Q.T)

    k_vec = np.concatenate(([[1]], np.zeros((r-1, 1))))
    q = - (IS.T * w.dot(vec(d)))[r:] +
    np.hstack(np.dot(Q[:r, r:].A.T, k_vec))


    zcons = -IS[:, r:]          # Enforce   -D <= 0
```

```
            ocons =  IS[:, r:]          # Enforce    D <= 1

            matrix_cons = sparse.bmat([ [zcons], [ocons] ])

            n0 = zcons.shape[0]
            n1 = ocons.shape[0]
            v_cons = np.concatenate(( np.zeros((n0, 1)) + IS[:, :r].dot(k_vec),
                                      np.ones((n1, 1))  - IS[:, :r].dot(k_vec)))

            e_results = cvxopt_qp(Q.A[r:, r:], q, matrix_cons.A,
                                    v_cons.flatten(), solver = 'mosek')
            _raise_status(e_results)
            e_vec = e_results[_vec_str]
            e = toarray(np.concatenate((k_vec, e_vec)), e.shape)

        except KeyboardInterrupt:
            print("Saving current iteration before quitting ...")
            np.savetxt(s_recovery, s)
            np.savetxt(e_recovery, e)
            raise

        # Update the previous iteration's chi-squared, calculate new
        chi2_prev = chi2
        delta_d = sparse.lil_matrix(vec(d) - vec(np.dot(s, e))).T
        chi2 = float((delta_d.T * w * delta_d).A)

        dtime = time.time() - _start_time
        print("Iteration %d; Chi-squared %.5e; Time %s" % (i, chi2, dtime))

        pdiff = np.abs(chi2 - chi2_prev) / chi2
        if pdiff < tol:
            print(_break)
            print("Stopped after iteration %d" % i)
            print("Chi-squared converged to %.5e" % chi2)
            status_code = True
            break

    else:
```

```
        print(_break)
        print("Reached maximum number of iterations")
        print("Final chi-squared: %f" % chi2)
        pdiff = np.abs(chi2 - chi2_prev) / chi2
        print("Final percent difference: %f%%" % pdiff)
        status_code = False

    #make True if want complementary effects
    s_final, e_final = decomp.matrix_decomposition(np.dot(s, e), k, False)
    d_final = np.matmul(s_final,e_final)
    print("size of matrix S is:", np.shape(s_final))
    print("size of matrix E is:", np.shape(e_final))
    print("size of matrix D is:", np.shape(d_final))
    if save:
        sfile = file_fmt % "s_rank_%s" % k
        efile = file_fmt % "e_rank_%s" % k
        dfile = file_fmt % "d_rank_%s" % k
        np.savetxt(sfile, s_final)
        np.savetxt(efile, e_final)
        np.savetxt(dfile, d_final)
        print("Saved files as", sfile, efile, sep = ", ")
    return (chi2, status_code, pdiff, i)

def main():
    print("working")
    """Command line interface for minimization.

    python quadsvole,py file rank [-f --fiducial]
    Args:
     *  file:       Path to noisy probabilities (uncertainty file is inferred
                        from this file)
     *  rank:       Rank of matrix to fit
    """
    _desc = "Perform weighted low rank fit on a noisy probability matrix"
    parser = argparse.ArgumentParser(description = _desc)
    parser.add_argument("file", help = "File path to probabilities")
    parser.add_argument("rank", help = "Rank of matrix to fit",
                                type = int, nargs = "+")
```

```
    parser.add_argument("-t", "--tol", help = "Chi-square tolerance",
                                type = float)
    args = parser.parse_args()
    for k in args.rank:
        results = low_rank_approx(args.file, k)
        chi2_qpfit, status, pdiff, iters = results
        print(75 * "-")
        print("Exit status code %s after %d iterations" % (status, iters))

if __name__ == '__main__':
    main()
```

A.2.1 Python Module for Modified QR Decomposition

The code used to perform the initial decomposition of D_k^{realized}. This code was originally written by Tim Hill and was adapted for the purposes of this thesis.

```
    """
Author: Tim Hill, Adapted by Michael Grabowecky
Module decomp.py

Module for decomposing the GPT of best fit (GOB).

Contents:
Function matrix_decomposition
    - Decomposes a matrix M into matrices S, E; M = SE
"""

import numpy as np

def matrix_decomposition(M, k, full_output = False):
    """matrix_decomposition performs the low-rank decomposition.

    matrix_decomposition decomposes a matrix M (m x n) into the matrices
    S (m x k) and E (k x n).

    Specifically, if full_output is specified, the complementary probabilities
```

```
  are added to M to increase the number of effect vectors we get. If M
  is shape (m, 1 + n), we make it shape (m, 1 + 2n) by adding n columns,
  where the jth additional column is 1 - M[:, j].

  Args:
   *  M: The probability matrix to decompose. Assumes it starts with
         a column of 1s
   *  k: The rank of the matrix (necessary if we also want to use
         this for the initial decompositioN)
   *  full_output = True: Option to add complementary effects (True) or
         not add complementary effects (False).

  Returns:
   * S: Matrix of states (m * k)
   * E: Matrix of effects
         full_output == False: (k * n)
         full_output == True:  (k * 2n - 1)
  Where (assuming M is actually rank k), M = SE
  """
  if full_output:
#       add complementary effects
     M = np.concatenate((M, 1 - M[:, 1:]), axis=1)

  # Calculate QR decomposition
  q, r = np.linalg.qr(M)

  # Remove unnecessary entries
  # Since M is only of rank k, there are only k non-zero columns in
  #  q and only k non-zero rows in r.
  # (If M is not rank k, this forces it to be rank k)

  q = q[:, :k]
  r = r[:k, :]

  sub_data = np.dot(q[:, 1:], r[1:])  # (m-1, k-1) subset of values
  u, s, v = np.linalg.svd(sub_data)
  u = u[:, :k-1]          # make u shape (m-1, k-1)
  s = np.diag(s[:k-1])    # make s a diagonal matrix (k-1, k-1)
```

```
    v = v[:k-1]             # make v shape (k-1, n-1)
    sub_p = np.dot(u, np.sqrt(s))
    sub_l = np.dot(np.sqrt(s), v)

    q[:, 1:] = sub_p
    r[1:] = sub_l

    transformation = np.ones(k)
    transformation[0] = 1./q[0,0]
    transformation = np.diag(transformation)

    pp = np.dot(q, transformation)
    ll = np.dot(np.linalg.inv(transformation), r)

    return pp, ll
```

A.3 Finding the Optimal transformation matrix $A_\star$

The Mathematica code used to find the optimal transformation matrix $A_\star$ (solution to Eq. (4.8)).

```
(*Author: Michael Grabowecky, QOQI 2021
This code finds the tranformation matrix R such that D = SRR^-1E = \
S'E'. The purpose is to find a particular decomposition that
reconstruct a set of states/effects that resemble the spaces \
predicted by qutrit QT*)
ClearAll["Global`*"]
(*import theoretical and experimental S and E matrices from the \
output of the QR decomposition*)
(*Theoretical S*)
S = Import[
   ToString[
    StringForm[
     "filepath"]],
   "Table", "FieldSeparators" -> " "];
(*Theoretical E*)
```

```
Et = Import[
   ToString[
    StringForm[
     "filepath"]],
   "Table", "FieldSeparators" -> " "];
(*Experimental S*)
S1 = Import[
   ToString[
    StringForm[
     "filepath"]], "Table", "FieldSeparators" -> " "];
(*Experimental E*)
E1 = Import[
   ToString[
    StringForm[
     "filepath"]], "Table", "FieldSeparators" -> " "];


(*add the unit effect to the theoretical E matrix*)
Et = Transpose[Et];
iden = {{1}, {0}, {0}, {0}, {0}, {0}, {0}, {0}, {0}};
Et = Join[iden, Et, 2];

(* check dimension consistency of S and E*)
Dimensions[S] == Dimensions[S1]
Dimensions[Et] == Dimensions[E1]

(*rank of the the matrix D, defines the size of transformation matrix \
R*)
k = 9;
(*define matrix of variables for optimization problem*)
mX = Array[X, {k, k}];

(*prepare matrices for optimization*)
S1R = S1.mX;
REt = Transpose[mX.Et];
E1trans = Transpose[E1];

(*Solve for the optimal transformation matrix R s.t S = S1R*)
```

```
Rsolve =
 NMinimize[
  Sum[Sum[(S[[i, j]] - S1R[[i, j]])^2, {j, 1, k}], {i, 1,
    Dimensions[S][[1]]}], Flatten[mX]]

(*retrieve matrix elements from solution*)
m = Table[Flatten[mX][[i]] /. Rsolve[[2]][[i]], {i, 1, k*k}];

(*construct the transformation matrix from the output of the \
optimization*)
R = ArrayReshape[m, {k, k}];
(* compute the inverse of R*)
Rinverse = Inverse[R];
MatrixForm[R]

(*determinant check (must be non-zero)*)
Det[R] != 0
Det[R]

(*calculate reconstructed S and E matrices*)
Srecon = S1.R;
Erecon = Rinverse.E1

(*the reconstructed S and E matrices*)
Srecon // MatrixForm;
Erecon // MatrixForm;

(*mean absolute difference between theoretical and reconstructed S \
and E matrices*)
Mean[Abs[Flatten[Srecon - S]]]
Mean[Abs[Flatten[Erecon - Et]]]
```

A.4 Calculating the Points of Intersection with the Realized and Consistent GPT Spaces

The Mathematica code for finding the points of intersection between random rays and the realized and consistent GPT state spaces. The functions that solve the linear programming problem $Mx = b, x \geq 0$ were originally written by Elie Wolfe.

```
(*Author: Michael Grabowecky, QOQI 2021
This code is used to compute the points of intersection between a set \
of random vectors and the experimentally realized and logically \
consistent GPT spaces*)
ClearAll["Global`*"]
(*These functions were witten by Elie Wolfe
they check for the existence of a solution to the linear program \
Mx=b, x \[GreaterEqual] 0
user calls function ExistsPosLinSol$Mathematica
WARNING: You must use the explicit variable name b when calling the \
function*)

BvecToEqualityVec[b_?VectorQ] :=
  ArrayPad[ArrayReshape[b, {Length[b], 1}], {{0, 0}, {0, 1}}];

ExistsPosLinSol$Mathematica[M_?MatrixQ, B_?MatrixQ,
   opts : OptionsPattern[LinearProgramming]] :=
  With[{objective = SparseArray[{}, Dimensions[M][[2]], 0.0]},
   Internal`HandlerBlock[{"Message",
     Switch[#, Hold[Message[LinearProgramming::lpsnf, ___], _],
       Throw[False], Hold[Message[LinearProgramming::lpsnfp, ___], _],
        Throw[False]] &},
    Quiet[(Check[
        Catch[LinearProgramming[objective, M, BvecToEqualityVec[b],
          opts]; True],
        False, {LinearProgramming::lpsnf,
         LinearProgramming::lpsnfp}]) /@
      B, {LinearProgramming::lpsnf}]]];

ExistsPosLinSol$Mathematica[M_?MatrixQ, b_?VectorQ,
```

```
   opts : OptionsPattern[LinearProgramming]] :=
  First[ExistsPosLinSol$Mathematica[M, {b}, opts]];

(*import experimentally realized GPT state vectors*)
RealizedStates =
  Import["filepath", "Table"];

 (*set number of random directions to find intersections for*)
 num = 1000;

 (***** REQUIRES HAAR-DISTRIBUTION ALGORITHM:
 Generate random directions using Haar-Distribution algorithm,
 convert result to Bloch vector representation,
 assign result to var name "directions"*****)

(*clear key variables, just in case*)
Clear[M, b, sol]

(*Make columns of M be the realized GPT states vectors*)
M = Transpose[RealizedStates];

(*Loop for computing intersection points*)
norms = AbsoluteTiming[
   Reap[Do[smallvec = Flatten[{1, 0.5*directions[[i]]}];
       largevec = Flatten[{1, 1.5*directions[[i]]}];
       While[True,
        (*find midpoint*)
        b = (smallvec + largevec)/2;
        (*call linear program solver*)
        sol =
         ToString[ExistsPosLinSol$Mathematica[M, b, Method -> "CLP"]];
        (*set either the long vector or the short vector equal to the \
        midpoint*)
        If[StringTake[sol, 5] == "False", largevec = b];
        If[StringTake[sol, 4] == "True", smallvec = b];
        (*check loop break threshold*)
        If[
         Norm[largevec[[2 ;; 9]]] - Norm[smallvec[[2 ;; 9]]] < 0.0001,
```

```
          Break[]];
        ];
       Sow[Norm[b[[2 ;; 9]]], {i, 1, num}]][[2]][[1]]];

(*returns the norms associated with the intersection points*)
norms

(* to compute intersections with S consistent,
run algorithm with consistent states imported instead*)
```